Service Architecture Leveraging Tuxedo (SALT)

Product Overview

12*c* Release 2 (12.2.2)

April 2016

SALT Overview

SALT Overview

The following sections provide an overview to the SALT product:

- Understanding SALT
- SALT Release History
- SALT Components
- SALT Use Cases
- Configuring Web Services with SALT
- SALT Supported Standards
- What’s Next?

Understanding SALT

Service Architecture Leveraging Tuxedo (SALT) is an add-on product option for Oracle Tuxedo, enabling Oracle Tuxedo applications to participate in SOA environments.

SALT allows external Web services applications to invoke Oracle Tuxedo services as Web services, and Oracle Tuxedo applications to invoke external Web services. SALT does not require any coding to achieve this.

Understanding SALT Web Services

SALT complies with standard Web service specifications (SOAP 1.1, SOAP 1.2, and WSDL 1.1), allowing SALT to interoperate with other Web service products and SALT Overview development toolkits. Oracle Tuxedo applications can easily integrate with Web services applications using SALT.

What Are Web Services?

Web services are a set of functions packaged into a single entity made available to other systems on a network. They can be shared and used as a component of distributed Web-based applications. The network can be a corporate intranet or the Internet. Other systems, such as customer relationship management (CRM) systems, order-processing systems, and other existing back-end applications, can call these functions to request data or perform an operation. Because Web services rely on standard technologies which most systems provide, they are an excellent means for connecting distributed systems together.

The software industry has evolved toward loosely coupled service-oriented applications that interact dynamically over the Web. The applications break down the larger software system into smaller modular components, or shared services. These services can reside on different computers and can be implemented by vastly different technologies. They are packaged and made accessible using standard Web protocols, such as XML and HTTP.

Web services share the following properties that make them easily accessible from heterogeneous environments:

- Web services are accessed using widely supported Web protocols such as HTTP.
- Web services describe themselves using an XML-based description language.

Web services communicate with clients (both end-user applications or other Web services) through simple XML messages that can be produced or parsed by virtually any programming environment or manually, if necessary.

Why Use SALT?

SALT is a native Oracle Tuxedo Web service integration solution. It reduces Oracle Tuxedo/Web service integration costs and decreases conversion processes that may exist with other solutions for accessing Oracle Tuxedo services. It enables seamless connectivity between Oracle Tuxedo applications and external Web service applications.

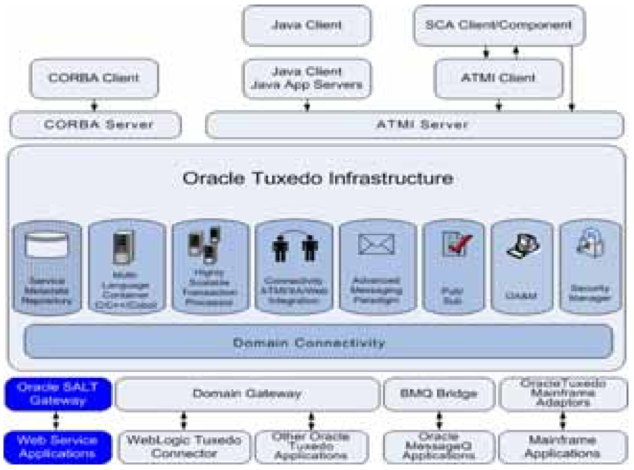
SALT allows existing Oracle Tuxedo services (inbound) to be easily exposed as Web services without additional programming tasks. It also allows you to create native Oracle Tuxedo applications that access external Web services (outbound) transparently.

Major Web services benefits include:

- Interoperability among distributed applications that span diverse hardware and software platforms
- Easy, widespread access to applications using Web protocols
- A cross-platform, cross-language data model (XML) that facilitates developing heterogeneous distributed applications

Figure 1 illustrates how the SALT gateway is used in the Oracle Tuxedo framework.

Figure 1 SALT Gateway/Oracle Tuxedo Infrastructure


Java Client
SCA Client/Component
CORBA Client
Java Client
Java App Servers
ATMI Client
CORBA Server
ATMI Server
Oracle Tuxedo Infrastructure
Service Metadata Repository
Multi Language Container C/C++/Cobol
Highly Scalable Transaction Processor
Connectivity ATMI/XA/Web Integration
Advanced Messaging Paradigm
Pub/Sub
/Q
Security Manager
Domain Connectivity
Oracle SALT Gateway
Domain Gateway
BMQ Bridge
OracleTuxedo Mainframe Adaptors
Web Service Applications
WebLogic Tuxedo Connector
Other Oracle Tuxedo Applications
Oracle MessageQ Applications
Mainframe Applications

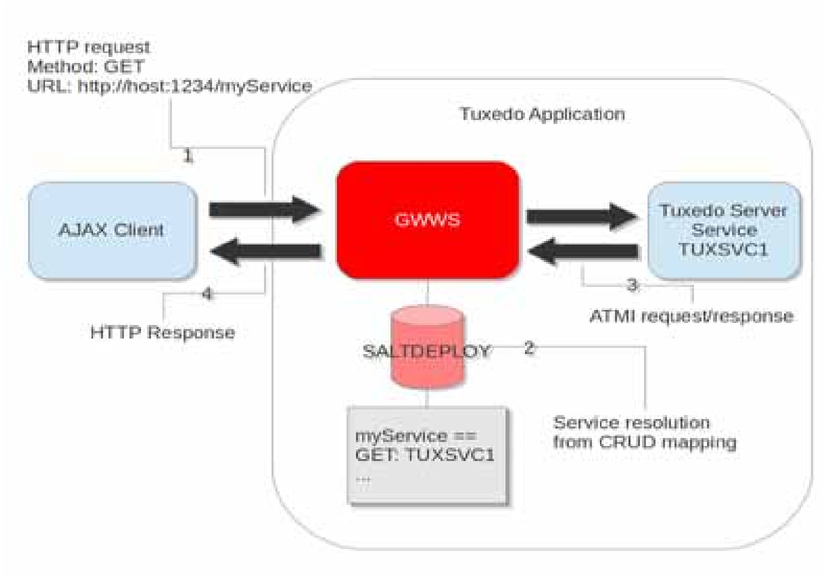
REpresentational State Transfer (REST) Support

SALT provides access to existing Tuxedo servers using a simplified HTTP format, following a principle more commonly known as REpresentational State Transfer (REST). In order to use REST, two new sections will need to be configured in `SALTDEPLOY`: HTTP/Network to specify the HTTP and/or HTTPS (SSL) port that the gateway will use to listen for REST requests, and HTTP/Services to specify which Tuxedo services are to be exposed and how; the combination of REST service name (arbitrary choice that will identify a REST servicein the calling URL) + HTTP method (`GET`/`POST`/`DELETE`/`PUT`) will map to the actual Tuxedo service so that CRUD principles can be expressed.

Once enabled, the following will be the behavior at runtime:

- A client program sends an HTTP message to GWWS. This message contains the verb (`GET`, `POST`, `PUT` or `DELETE`), headers (content-type, etc.) and URL.
- "GWWS uses the verb to determine what to do with data: for `GET` or `DELETE`, the query string is used to construct a request for the actual Tuxedo service, for `POST` or `PUT`, the post data is used.
- The URL part of the request is used to determine the service to be called: for example, a call such as '`POST` http://host:1234/myService' results in GWWS assembling a request for the Tuxedo service mapped to by the 'myService' name in `SALTDEPLOY` (e.g., '`myService`' with `GET` method may map to a Tuxedo service named `BALANCE`), converting the reply from FML32 to JSON (assuming service returns FML32) and sending it back to the caller.

Figure 2 REST in GWWS Architecture



SALT Release History

SALT is the latest add-on to the Oracle Tuxedo product family. Developed in 2006, SALT is designed to provide a seamless Oracle Tuxedo solution of integrating Oracle Tuxedo applications and standard Web services application.

Release 1.1

Release 1.1 is the initial SALT release. Made available in 2006, SALT 1.1 introduced the following major features:

- Inbound Service Support
  Permits Web service applications to invoke native Oracle Tuxedo services
- HTTP and HTTP over SSL Transport Support
- Asynchronous and Reliable Messaging Support

Release 2.0

The SALT 2.0 release incorporates significant enhancements based on the SALT 1.1 release. SALT 2.0 introduced the following features:

- Outbound Service Support
- Extended WS-* Standards Support
- SOAP Message Transmission Optimization Mechanism Support (MTOM)
- Oracle Tuxedo TPFAIL Support for Web Services
- Extensible Data Type Mapping and Message Conversion
- Multiple Encoding Support

- Configuration-Driven Deployment
- Leveraging the Oracle Tuxedo Service Metadata Repository
- Data Type Mapping and Message Conversion
- Asynchronous and Reliable Messaging
- Web Service Security Support

Release 10*g* R3 (10.3)

SALT 10*g* Release 3 (10.3) introduced the following features:

- Service Component Architecture (SCA) Programming

  SCA provides a new programming model that aims at simplifying component re-use and seamless communications between components. The SALT 10*g* Release 3 (10.3) SCA container enables new programming model and leverages the most valued Oracle Tuxedo features (such as reliability, availability, scalability, and performance). SALT 10*g* Release 3 (10.3) introduces the following SCA features:

  – Client-side binding for SCA invocations over ATMI and SOAP

  – Server-side binding for serving SCA requests made over ATMI and SOAP

  – Client-side binding for SCA invocations from Java environments

  – Development and runtime tools: Commands to build and deploy SCA clients and servers as well as commands for runtime administration. For more information, see the SALT 10g Release 3 (10.3) Command Reference Guide.

  – Authentication and authorization for SCA services

  – Global transactions

  – Thread-safe SCA/SDO clients and servers

  – SCDL schema validation

  – Support for simple data types

  – Support for complex data types using SDO

  – Automatic data transformation to/from Oracle Tuxedo buffer types

  – Support for multi-byte characters using multiple encoding

- Service Contract Discovery

Automatically discover service contract information at run time. The generated information can be put into metadata repository automatically or to a file which can then be loaded manually into the metadata repository using the `tmloadrepos` utility. For more information, see Configuring a SALT Application in the SALT *Administration Guide*.

- Access Log for All Incoming Requests

  Assists Oracle Tuxedo client administrators to monitor application validity at runtime. You can record application high water client count, current client count, and named users.

Release 11*g* R1 (11.1.1.1.0)

SALT Release 11g R1 (11.1.1.1.0) introduced the following features:

- Python and Ruby Support

  Python and Ruby support in SALT SCA provides a simple SCA client API for each language to perform SCA calls from Python or Ruby client programs, and an SCAHOST system server to allow access to Python or Ruby components. Python, Ruby or C++ components can now call or be called to/from other Python, Ruby or C++ components.

  For more information, see *SALT SCA Programming* in the SALT Programming guide.

- SCA Structure support

  Provides additional C++ structure functionality for improved performance.

  For more information, see *SALT SCA Programming* in the SALT Programming guide.

- Scatuxgen Metadata Generation Tool

  The Scatuxgen Metadata Generation Tool parses C++ interfaces as used to develop SCA components for the SCA runtime functionality introduced in SALT 10gR3. It generates Oracle Tuxedo metadata repository interface data, and optionally a Web Services Definition File (WSDF) document.

  For more information, see the SALT Command Reference Guide.

- WS-TX Support

  Provides bi-directional transactional interoperability between Web Services and Oracle Tuxedo applications. Applications transparently make use of the GWWS system server transactional capabilities.

  Outbound XA transactions are transparently propagated outside of an Oracle Tuxedo domain and mapped one-to-one to WS-TX transactions. Inbound WS-TX transactions are propagated into an Oracle Tuxedo domain and mapped to XA transactions.

For more information see, *Configuring WS_TX Support* in the SALT Configuration guide and WS-TX Interoperability in SALT Interoperability.

Release 11*g* R1 (11.1.1.2.0)

SALT 11*g* Release 11g R1 (11.1.1.2.0) introduced the following features:

- SOCKS Proxy Support

  SOCKS proxy support provides a configurable element associated to outbound endpoints and allows configuring a SOCKS server address that will proxy outbound connections according to the SOCKS V4 and V5 protocol.

  Inbound support is not be necessary as SOCKS proxies incoming connections without listening endpoints being aware of it. For more information, see the SALT Deployment File Reference in the SALT Reference Guide.

  The `wsdlcvt` command is enhanced to connect to a SOCKS proxy server. For more information, see the SALT Command Reference in the SALT Reference Guide.

Release 11*g* R1 (11.1.1.2.2)

SALT 11*g* Release 11g R1 (11.1.1.2.2) introduces the following features:

- Apache 2 Web Server Plug-in Support

  Web Server plug-ins in Apache 2 format, for interfacing Apache HTTP server 2, Oracle HTTP Server or iPlanet Server and Oracle Tuxedo-based web applications.

- Web Gateway Interface Support

  Support of a Web Gateway Interface in order to develop Oracle Tuxedo services that generate dynamic HTML pages.

- PHP, Python and Ruby Web Scripts Support
  - PHP scripts supported directly
  - Python scripts supported via WSGI
  - Ruby scripts supported via Rack
  - PHP, Python and Ruby applications using most web frameworks such as Zend framework, Symfony, Django or Rails run as-is in an Oracle Tuxedo environment.

- PHP SCA Script Support

  Support for PHP scripts as SCA client and SCA components, for SOA integration.

For more information, see the SALT Administration Guide.

Release 12*c* R1 (12.1.1)

SALT 12*c* Release 12c R1 (12.1.1) introduces the following features:

- Web Services Configuration Tool

  SALT 12c provides an HTTP-based configuration tool to expose existing Oracle Tuxedo services as Web services without manually editing configuration files. It allows you to learn service definitions for existing Oracle Tuxedo services, edit service definitions in metadata repository, and create Web services definitions and `SALTDEPLOY` files through an easy to use graphical user interface.

  For more information, see Enabling the SALT Configuration Tool in the SALT Configuration Guide.

- Security Assertion Markup Language (SAML) Single Sign-On (SSO) Support

  Provides the ability to recognize an SAML token inside a SOAP message request received by the SALT Web Services Gateway (GWWS). Based on the token contents, GWWS makes the decision to grant or deny access to Oracle Tuxedo resources.

  For more information, see Configuring Oracle Tuxedo Web Services/Configuring Security Features in the SALT Configuration Guide.

- New Data Type Support

  Supports nested View32 data types. It also supports mapping of additional View and View32 primitive types to and from XML.

  For more information, see Data Type Mapping and Message Conversion in the SALT Programming Guide.

Note: SALT Python, Ruby, and SCA features are now part of Oracle Tuxedo 12c Release 1 (12.1.1). For more information, see Oracle Tuxedo 12c Release (12.1.1) documentation.

Release 12*c* R2 (12.1.3)

SALT 12*c* Release 2 (12.1.3) introduces the following features:

- RESTful Web API

  Existing Oracle Tuxedo services can be made available to and accessed by http clients as RESTful Web services eliminating the need to use SOAP/http for lightweight applications and expediting integration with other applications. RESTful web API can use XML or JSON payload for data transfer.

In addition to accessing Oracle Tuxedo services as RESTful Web services, Oracle Tuxedo applications can also access external RESTful services without having to write any code. Oracle Tuxedo applications can invoke RESTful services the same as if invoking Oracle Tuxedo services. The SALT gateway acts as the proxy for RESTful Web services.

For more information, see *Enabling the SALT Configuration Tool, REpresentational State Transfer (REST) Option* in the SALT Configuration Guide.

- Custom HTTP Headers

  HTTP headers can pass relevant application control information to or from Oracle Tuxedo services. For incoming RESTful Web services, any custom HTTP header is attached to an Oracle Tuxedo buffer and passed to the invoked Oracle Tuxedo service. An Oracle Tuxedo service header can be read using a provided API. Similarly, an Oracle Tuxedo application can set HTTP headers in an Oracle Tuxedo buffer using a provided API, which in turn is converted to an HTTP header by the SALT gateway.

  For more information, see *Enabling the SALT Configuration Tool, Custom HTTP Headers* in the SALT Configuration Guide.

- WS-Security for External Web Services

  Message-level authentication is provided using an X.509 certificate to sign messages. Oracle Tuxedo can invoke an external Web service using SOAP/http with the principal identity of the X.509 certificate.

  For more information, see *Configuring Oracle Tuxedo Web Services/Configuring Security Features* in the SALT Configuration Guide.

- Data Transformation Tracing

  Tracing all incoming and outgoing messages is enabled (including RESTful Web services, SOAP/http Web services, and all data transformation from XML to Oracle Tuxedo buffers and vice-versa).

  For more information, see *XML-to-Tuxedo Data Type Mapping for External Web Services* in Data Type Mapping and Message Conversion Services of the SALT Programming Guide.

- ECID Propagation

  ECID (Execution Context ID) is propagated with each request within Oracle Tuxedo and across various products in an Oracle stack. ECID propagation enables request correlation across Oracle Tuxedo domains and Oracle products (such as Oracle WebLogic Server, Oracle Database, etc.), making it quicker to diagnose application problems.

For more information, see Configuring Tuxedo for Propagating ECID.

- Dynamic Configuration and MIB

  You can dynamically reload configuration file changes without any downtime. An MIB interface is provided, which enables reading Web services configuration and runtime statistics.

  For more information, see *Enabling the SALT Configuration Tool, MIB Class Interface* in the SALT Configuration Guide.

- XML Complex Attribute Mapping

  Enhanced usability is provided by using an attribute field within a `complexType` element in a WSDL for accessing external Web services. `complexType` attribute fields are mapped "one-by-one" to corresponding FML32 fields.

  For more information, see *Data Type Mapping and Message Conversion* in the SALT Programming Guide.

- Configuration Tool Enhancements

  The configuration tool has been enhanced to provide the following features:

  – Support for RESTful Web services

  – Enables import of external Web services

  – Support for FireFox and Safari Web browsers

  – A test client for RESTful Web services

  For more information, see *The* SALT *Configuration Tool* in the SALT Configuration Guide.

Release 12*c* R2 (12.2.2)

SALT 12*c* Release 2 (12.2.2) introduces the following features:

- Custom SOAP Headers

  This feature enables use application specific headers in SOAP messages when accessing Tuxedo services using SOAP/http. This feature leverages the existing `tpsetcallinfo()` and `tpgetcallinfo()` APIs to place/retrieve data into/from the SOAP header for inbound/outbound directions.

  - Inbound (Oracle Tuxedo services exposed as web services, or native web services).
  - Outbound (invoking web services as Oracle Tuxedo services, or external web services).

  For more information, see `tpsetcallinfo()`, `tpgetcallinfo()`, *Enabling the SALT Configuration Tool*, *Custom SOAP Headers* in the SALT Configuration Guide.

- RECORD Buffer Type Support

  `RECORD` buffer type is now supported in SALT. `RECORD` buffer type is a flexible way to correctly and completely represent COBOL copybook records.

  `RECORD` buffers are used in SALT to expose mainframe transactions and ART Transactions as Web services and to access Web services from these transactions.

  For more information, see *Using Oracle Tuxedo Service Metadata Repository for SALT*, and *Tuxedo-to-XML Data Type Mapping for Oracle Tuxedo Services*.

- Mainframe Transaction Publisher for SALT

  This release of SALT makes it easier for transactions running in IBM mainframe's CICS/IMS environments to be exported as Web services (SOAP/http or RESTful Web services). In addition, such mainframe transactions can also access external Web services using SOAP/http or RESTful API. Such integration is completely configuration driven and no code development is needed.

  SALT includes tools, such as `wscobolcvt`, which allow to import COBOL copybook and create metadata repository artifacts, WSDL and other required configuration to expose mainframe transactions as Web services. Similarly, tools allow import of a WSDL for external Web service and create required artifacts for mainframe transactions to access external Web services.

  For more information see, `wscobolcvt`, *SALT Mainframe Transaction Publisher*, and *XML-to-Tuxedo Data Type Mapping for External Web Services*.

- New Tuxedo Services Console

  - This release of SALT replaces existing SALT configuration tool with a new Tuxedo Services Console to export, import Tuxedo services. This new console provides following major functionality:
  - Services Management: add/edit/delete Tuxedo service definitions in the Tuxedo metadata repository.
  - Configure Web services (SOAP and REST): enables Oracle Tuxedo services to be accessed as SOAP or REST services and enables Tuxedo applications to access external SOAP/REST Web services.

  - Integrate with mainframe transactions: enables mainframe transactions to be accessed as Web services (SOAP or REST), or enables mainframe transactions to access external Web services.

  For more information, see `TMADMSVR` and `MTP`.

- Single-Sign-on with Oracle Access Manager

  This release of SALT integrates with Oracle Access Manager (OAM) seamlessly, with automatic detection of the OAM authentication server configuration and handling of OAM tokens. This achieves single sign-on at the HTTP level for support of secure SOAP and REST access.

  For more information see *Configuring Oracle Tuxedo Web Services*, Configuring Security Features.

SALT Components

SALT consists of the following major components:

- SALT Gateway (GWWS)
- WSDL Assistant Utilities

SALT Gateway (GWWS)

The SALT provided Oracle Tuxedo system server (GWWS), connects with other Web service applications via SOAP over HTTP/S protocol. The GWWS server acts as an Oracle Tuxedo gateway process and is managed in the same manner as general Oracle Tuxedo system servers. Each GWWS server has bi-directional (inbound/outbound) capability. The GWWS server:

- accepts SOAP requests from Web service applications and issue Oracle Tuxedo native calls to Oracle Tuxedo services.
- accepts Oracle Tuxedo ATMI requests and issues SOAP calls to Web Service applications.

You can have multiple GWWS instances in one Oracle Tuxedo domain. The same functionality for multiple GWWS instances is provided by specifying the same SALT configuration to improve throughput and failover protection. You can also group multiple GWWS instances in different configuration files for different purposes.

When the GWWS server boots, it loads the specified SALT configuration file and Oracle Tuxedo service contract information from the Oracle Tuxedo Service Metadata Repository.

The GWWS server also acts as a simple HTTP Web server for WSDL document and XML Schema file download.

WSDL Assistant Utilities

The Web Services Description Language (WSDL) is an XML-based specification that describes a Web service. A WSDL document describes Web service operations, input and output parameters, and how a client application connects to the Web service. SALT provides two utilities (`tmwsdlgen` and `wsdlcvt`) to map Oracle Tuxedo applications and Web Service WSDL descriptions.

WSDL Generator from Oracle Tuxedo Definitions

When using SALT to publish Oracle Tuxedo services as Web services, you do not need to compose a WSDL document manually; it is automatically generated as part of the SALT Web service development process. The generated WSDL document can be integrated using Web service development tools, or can be published to a UDDI server.

There are two ways to obtain a WSDL document:

- Use `tmwsdlgen` (the WSDL document file generating utility).
- Download the GWWS server generated WSDL document via HTTP(S).

WSDL Convertor to Oracle Tuxedo Definitions

To support external Web Service applications, external WSDL documents need to be converted. The SALT conversion utility, `wsdlcvt`, converts external WSDL documents to Oracle Tuxedo specific definition files (SALT Web Service Definition file, Oracle Tuxedo Service Metadata Repository Definition file and FML32 Field Table Definition file).

The SALT Web Service Definition file can be imported into a SALT Deployment file and utilized by a particular GWWS server. The Oracle Tuxedo Service Metadata Repository Definition file and FML32 Field Table Definition file provide service interface descriptions for Oracle Tuxedo client programming.

SALT Use Cases

The following sections describe the most common SALT Web services use cases:

- Use Case 1: Exposing Native Oracle Tuxedo Services as Web Services
- Use Case 2: Invoking Web Services from Oracle Tuxedo Applications

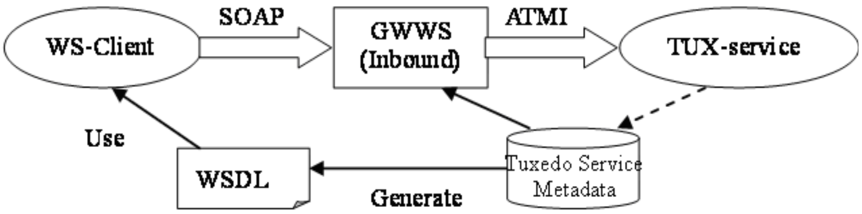
- Use Case 3: Connecting Oracle Tuxedo Domains Using SOAP Protocol
- Use Case 4: SAML Single Sign-On

Use Case 1: Exposing Native Oracle Tuxedo Services as Web Services

Native Oracle Tuxedo services can be exposed as Web services using standard Web service SOAP protocol. The GWWS server accepts SOAP requests through HTTP/S and then converts them into Oracle Tuxedo ATMI calls. SALT generates a WSDL document that describes the open standard Web service interfaces for Oracle Tuxedo services.The Oracle Tuxedo Service Metadata Repository is used to define Oracle Tuxedo service contract information. This is an “inbound” use case.

Figure 3 illustrates a generic inbound Web service call.

Figure 3 Exposing Oracle Tuxedo Services as Web Services (Inbound)



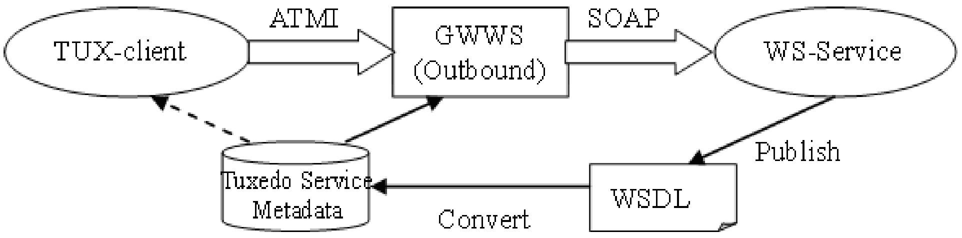
Use Case 2: Invoking Web Services from Oracle Tuxedo Applications

Web service applications can be imported into an Oracle Tuxedo domain, advertised as Oracle Tuxedo services through the GWWS server, and invoked from Oracle Tuxedo applications. SALT converts and maps each `wsdl:operation` as a particular Oracle Tuxedo service. The GWWS server advertises the mapped services (called SALT proxy services), and accepts Oracle Tuxedo ATMI requests from Oracle Tuxedo applications.

The Oracle Tuxedo Service Metadata Repository is used to store converted Oracle Tuxedo service contract information and helps Oracle Tuxedo programmers understand what type of Oracle Tuxedo buffers are expected for the imported SALT proxy services. This is an “outbound” use case.

Figure 4 illustrates a generic outbound Web service call.

Figure 4 Invoking Web Services from Oracle Tuxedo Applications (Outbound)



Use Case 3: Connecting Oracle Tuxedo Domains Using SOAP Protocol

SALT also allows you to connect two *different* Oracle Tuxedo domains using GWWS servers as an alternative to using /T domain. The GWWS server in the calling domain works in an outbound direction, the GWWS server in the receiving domain works in an inbound direction.

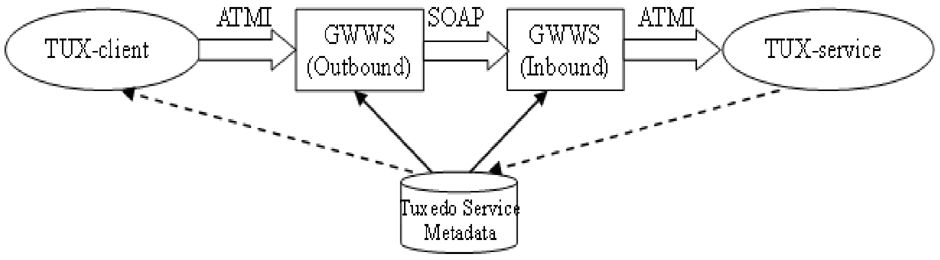
The receiving Oracle Tuxedo domain must propagate the Oracle Tuxedo service definition to the calling Oracle Tuxedo domain. This means that the calling domain Oracle Tuxedo Service Metadata Repository must contain the Oracle Tuxedo service definition file that runs in the receiving domain.

Note: This should be set up *manually*. The Oracle Tuxedo Service Metadata Repository infrastructure does not currently provide automatic propagation between Oracle Tuxedo domains.

The WSDL document is not required. SALT provides simple configurations to allow two GWWS servers to work together for domain connectivity using SOAP protocol without needing to exchange WSDL documents.

Figure 5 illustrates how to use SALT to connect two domains.

Figure 5 Connecting Two Oracle Tuxedo Domains with SOAP protocol



Two GWWS servers should not be used to create connections within the *same* Oracle Tuxedo domain, see Figure 6. Also, a single GWWS server cannot connect to itself, see Figure 7.

In either scenario, the GWWS server advertises the same Oracle Tuxedo services which are already advertised by other application servers. This might result in *dead-loop* service dispatching.

WARNING: It is strongly advised that you carefully plan and configure your SALT application to avoid these scenarios.

Figure 6 Two GWWS Servers Making a Connection Within the Same Oracle Tuxedo Domain

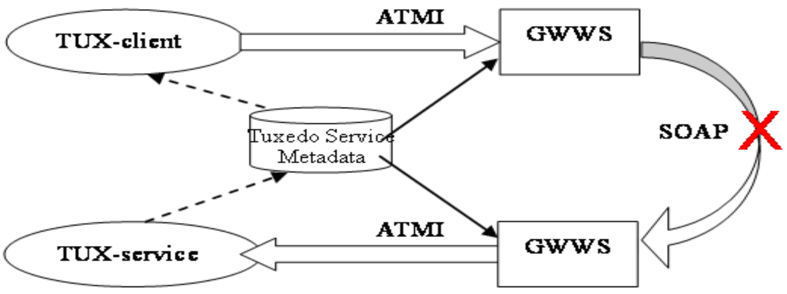
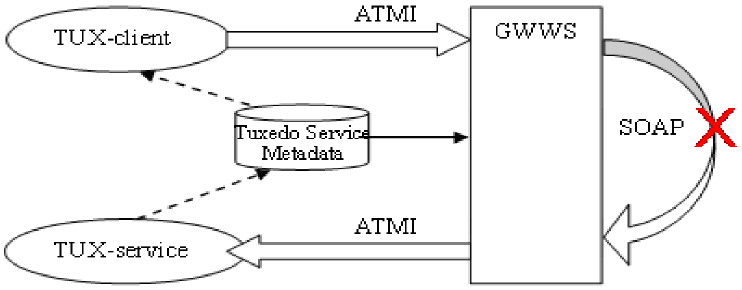


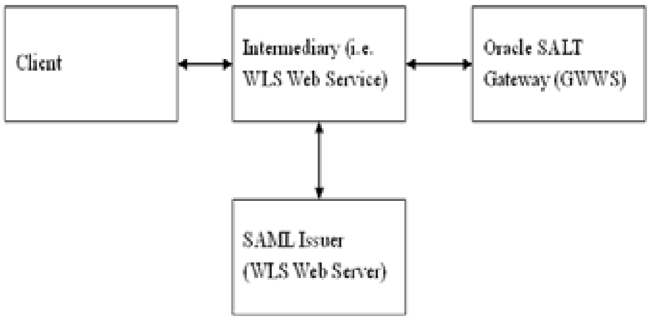
Figure 7 Single GWWS Server Making a Connection to Itself



Use Case 4: SAML Single Sign-On

The Web Service client user agent, usually a web browser, attaches a Security token with valid SAML assertions to a SOAP message using WSS: SOAP Message Security by placing assertion elements inside a `<wsse:Security>` header and sending the request to GWWSas shown in Figure 8

Figure 8 SAML Single Sign-On


Client
Intermediary (i.e. WLS Web Service)
Oracle SALT Gateway (GWWS)
SAML Issuer (WLS Web Server)

SALT will not accept these assertions unless it has successfully validated the integrity of the assertions. The only supported subject confirmation method is currently sender-vouches, in this case SALT must also have TRUST relationship with the attesting entity. When all these requirements are satisfied, the user is granted access to Oracle Tuxedo resources.

An Oracle WebLogic Server instance security can be a SAML issuer. In a WLS environment the typical intermediary is Web Service, but a web browser can also be an intermediary. To establish a TRUST relationship with an attesting entity, GWWS must have access to either a shared secret with attesting entity or its public key.

The SAML issuer is the party creating the SAML assertion, and "intermediary" is the party that attaches the SAML security token to the SOAP message before it is forwarded to GWWS.

The general flow is as follows:

1. Client sends request to WLS.
2. WLS authenticates the client.
3. WLS forwards all necessary login information to the issuer (WLS Web Server).
4. WLS Web Server issuer creates the SAML assertions and returns to WLS.
5. WLS attaches the SAML security token with SAML assertion to the SOAP message and forwards it to GWWS.
6. GWWS returns the reply/fault to WLS.
7. WLS returns the reply/fault to client.

There is a trust relationship between GWWS and WLS/issuer.

Configuring Web Services with SALT

The following steps are used typically when you configure Web services using SALT:

1. Configure Inbound Oracle Tuxedo Services
   a. Define Oracle Tuxedo application services using the Oracle Tuxedo Service Metadata Repository.
   b. Compose one or more SALT Web Service Definition Files (WSDF).
2. Configure Outbound Web Services
   a. Convert an external WSDL document into Oracle Tuxedo the following components: SALT Web Service Definition file, Oracle Tuxedo Service Metadata Repository Definition file, and FML32 Field Table Definition file.
   b. Resolve potential naming conflicts for the auto-generated service names and FML32 field names.

3. Load all Oracle Tuxedo Service Definitions into the Oracle Tuxedo Service Metadata Repository using `tmloaderpos`.
4. Compose the SALT Deployment File for both inbound and outbound services.
5. Add the `TMMETADATA` and `GWWS` servers to your Oracle Tuxedo `UBBCONFIG` file.
6. Boot the Oracle Tuxedo application.

Invoking Oracle Tuxedo Services Using Web Service Client Toolkits

1. Client end user downloads the WSDL document file from the GWWS server.
2. Client end user generates client-side stubcode from the WSDL document file with a SOAP development kit.
3. Generate client-side program.
4. Run the client to invoke the Web service with SOAP messages.

Invoking Web Services Using Oracle Tuxedo Programming Interfaces

1. Create an Oracle Tuxedo client/server program according to the generated Oracle Tuxedo Service Metadata Definition file. The client program can be written in any Oracle Tuxedo supported client-side programming language (C/C++, Java, COBOL, .NET, and so on).
2. Compile and deploy the Oracle Tuxedo client/server program.
3. Run the Oracle Tuxedo application to invoke the external Web service applications.

SALT Supported Standards

SALT support the following standards:

- SOAP Standards
- Single Sign-On Standards

SOAP Standards

- Standards for transmitting data and Web service invocation calls between the Web service and the user of the Web service.
  - SOAP 1.1

For more information, see:
http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

  - SOAP 1.2

    For more information, see:
    http://www.w3.org/TR/soap12-part0/

  - SOAP with Attachment

    For more information, see:
    http://www.w3.org/TR/SOAP-attachments

  - MTOM

    For more information, see:
    http://www.w3.org/TR/soap12-mtom/

- A standard for client applications to find a registered Web service and to register a Web service.
  - UDDI 2.0

    For more information, see:
    http://uddi.org/pubs/ProgrammersAPI-V2.04-Published-20020719.htm

- Standards for describing the Web service to clients so they can invoke it.
  - WSDL 1.1

    For more information, see:
    http://www.w3.org/TR/2001/NOTE-wsdl-20010315

  - WS-Policy

    For more information, see:
    http://specs.xmlsoap.org/ws/2004/09/policy/ws-policy.pdf

    http://specs.xmlsoap.org/ws/2004/09/policy/ws-policyattachment.pdf

- Standards for Web Service infrastructure.
  - WS-Addressing

    For more information, see:
    http://www.w3.org/Submission/2004/SUBM-ws-addressing-20040810/

  - WS-ReliableMessaging

For more information, see:
http://specs.xmlsoap.org/ws/2005/02/rm/ws-reliablemessaging.pdf

http://specs.xmlsoap.org/ws/2005/02/rm/WS-RMPolicy.pdf

- Standards for Web Service Security.
  - WS-Security 1.0

    For more information, see:

    http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf
    http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0.pdf
    http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0.pdf
  - WS-Security 1.1

    For more information, see:

    http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf
    http://www.oasis-open.org/committees/download.php/16782/wss-v1.1-spec-os-UsernameTokenProfile.pdf
    http://www.oasis-open.org/committees/download.php/16785/wss-v1.1-spec-os-x509TokenProfile.pdf
- Standards for Web Service Transactions.
  - WS-Coordination 1.0

    For more information, see:

    http://schemas.xmlsoap.org/ws/2004/10/wscoor/
  - WS-AtomicTransaction 1.0

    For more information, see:

    http://schemas.xmlsoap.org/ws/2004/10/wsat/

Single Sign-On Standards

- Single Sign-On Standards
  - SAML 1.1

    For more information, see: http://www.oasis-open.org/standards#samlv1.1.

– SAML 2.0

For more information, see: http://www.oasis-open.org/standards#samlv2.0.

What’s Next?

After becoming familiar with the SALT Product Overview, refer to the following topics for installing, configuring, and running Web services using the SALT product:

- Install the SALT product.

  Note: SALT 12c Release 2 Installation is now part of Oracle Tuxedo 12c Release 2 Installation. SALT 12c Release 2 supports the same platforms as Oracle Tuxedo 12c Release 2.

  For an explanation of how to install the product, refer to the *Oracle Tuxedo Installation Guide*.

- Administer the SALT product.
- For an explanation of how to configure and administer the product, refer to the *SALT Administration Guide*.
- Configure the SALT product.

  For an explanation of how to configure and administer the product, refer to the *SALT Configuration Guide*.

- Program Web Services using the SALT.

  For an explanation of how to program with SALT, refer to the *SALT Programming Guide*.

- SALT Web service samples

  For SALT Web service application samples, refer to the *SALT Sample Guide.*