# Contents

**Preface**

- Audience v
- Documentation Accessibility v
- Related Resources v
- Conventions vi

**1 Understand the SOAP Adapter**

- SOAP Adapter Capabilities 1-1
- What Application Version Is Supported? 1-3
- Workflow to Create and Add a SOAP Adapter Connection to an Integration 1-3

**2 SOAP Adapter Concepts**

**3 Create a SOAP Adapter Connection**

- Prerequisites for Creating a Connection 3-1
- Create a Connection 3-1
  - Add a Contact Email 3-2
  - Configure Connection Properties 3-3
  - Configure Connection Security 3-4
  - Configure an Agent Group 3-6
  - Test the Connection 3-7
- Upload an SSL Certificate 3-7

**4 Add the SOAP Adapter Connection to an Integration**

- Basic Info Page 4-1
- Trigger Operation Page 4-2
- Trigger Callback Operation Page 4-2
- Invoke Operation Page 4-3
- Header Page 4-3
Implement Common Patterns Using the SOAP Adapter

5-1 Configure MTOM Support in the SOAP Adapter
5-5 Consume Taleo SOAP APIs
5-5 Invoke a SOAP-Based Integration with a Timestamp
5-6 Configure Custom SOAP Headers for the SOAP Adapter
5-8 Call an Oracle Fusion Applications Business Intelligence Publisher Report Synchronously

Troubleshoot the SOAP Adapter

6-1 Regenerate the SOAP Adapter Connection After WSDL Definition Updates
6-2 Use the Correct SoapUI Version to Load and Test SOAP Endpoints
6-2 Edit Adapter Connections in Active Integrations
6-3 Multiple SOAP Adapter Invokes with Custom Headers Created Prior to 17.4.3
6-4 SOAP Endpoint Invocation Fails with OSB-380001: mustUnderstand Error
6-4 Specify Connection Property Values with the REST API
6-5 Callback Integrations Fail with a Configured SOAP Action Mismatch Error
6-6 Integrations Fail with SAML Security Policy Selected in Inbound Direction
6-6 Schemas Not Successfully Loaded in Mapper When Using Headers Configured with WSDLs Ending in asmx
6-6 Resolve SOAP Action Mismatch Errors in the SoapUI
6-7 Resolve Exceptions While Invoking Oracle Integration From External Clients
6-8 Connection Error When Using the Incorrect TLS Version
6-9 Extra Information is Included in the Response Headers Returned as Part of the Response Message
6-10 Basic Authentication Fields are Unavailable when Updating a SOAP Adapter Connection after Importing a 16.2.5 Integration
6-10 Unexpected Use of the Suppression Insertion of Timestamp into WS-Security Header Feature in the SOAP Adapter Causes an Unrelated Error Response
6-10 Pre-17.2.5 Integration Failures with an Invoke SOAP Adapter Configured with the Basic Authentication Security Policy and Suppress Timestamp Set to No
Preface

This guide describes how to configure the SOAP Adapter as a connection in an integration in Oracle Integration.

Note:
The information in this guide applies to all of your Oracle Integration instances. It doesn’t matter which edition you’re using, what features you have, or who manages your cloud environment. You’ll find what you need here, including notes about any differences between the various flavors of Oracle Integration when necessary.

Topics

• Audience
• Documentation Accessibility
• Related Resources
• Conventions

Audience

This guide is intended for developers who want to use the SOAP Adapter in integrations in Oracle Integration.

Documentation Accessibility

For information about Oracle’s commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Resources

See these Oracle resources:
• Oracle Cloud
  http://cloud.oracle.com
• Using Integrations in Oracle Integration
• Using the Oracle Mapper with Oracle Integration

Conventions

The following text conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>boldface</strong></td>
<td>Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.</td>
</tr>
<tr>
<td><em>italic</em></td>
<td>Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.</td>
</tr>
<tr>
<td><code>monospace</code></td>
<td>Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.</td>
</tr>
</tbody>
</table>
Understand the SOAP Adapter

Review the following conceptual topics to learn about the SOAP Adapter and how to use it as a connection in integrations in Oracle Integration. A typical workflow of adapter and integration tasks is also provided.

Topics:
• SOAP Adapter Capabilities
• What Application Version Is Supported?
• Workflow to Create and Add a SOAP Adapter Connection to an Integration

SOAP Adapter Capabilities

The SOAP Adapter can consume an external SOAP API in an integration in Oracle Integration. The message received from Oracle Integration can be passed as payload to an external SOAP endpoint by the SOAP Adapter. Any response received from the endpoint can be sent to the next action in the integration for further processing.

The SOAP Adapter can expose inbound SOAP endpoints for accepting SOAP requests that are addressed to a specific URI. The request body is passed to the next activity present in the integration as the message payload, along with the SOAP and HTTP headers.

Note:

The SOAP Adapter treats all endpoints as they are exposed. The SOAP Adapter does not filter or change any of the APIs exposed by the application to which you are connecting. If there is a native adapter for the application to which you are connecting, use that adapter instead. If you choose to use the SOAP Adapter instead of the native adapter, the API restrictions and deprecation policies apply as specified in the respective application's documentation. To connect to the Oracle HCM Cloud SOAP APIs, see Oracle HCM Cloud Adapter Capabilities.

The SOAP Adapter provides the following capabilities:
• SOAP Adapter capabilities when configured as a trigger:

Important:

Integrations exposed as SOAP APIs (using a SOAP Adapter-specific connection configured as a trigger) cannot accept attachments.
ENSURES THAT AN INCOMING STRUCTURED PAYLOAD (XML) FROM A CLIENT DOES NOT EXCEED 10 MB IN SIZE. IF THE SIZE OF THE PAYLOAD EXCEEDS 10 MB, AN HTTP ERROR CODE MESSAGE IS RETURNED TO THE CLIENT.

ENSURES THAT AN INCOMING UNSTRUCTURED PAYLOAD (MTOM) FROM A CLIENT DOES NOT EXCEED 512 MB IN SIZE. IF THE SIZE OF THE PAYLOAD EXCEEDS 512 MB, AN HTTP ERROR CODE MESSAGE IS RETURNED TO THE CLIENT.

ALLOWS CONFIGURING ONLY HTTPS PROTOCOL-BASED SOAP ENDPOINTS FOR ACCEPTING INCOMING SOAP REQUESTS.


SUPPORTS ACCESSING OF STANDARD AND CUSTOM SOAP/HTTP HEADER PROPERTIES PRESENT IN THE INCOMING SOAP REQUEST AND MAKING THEM AVAILABLE AS PART OF AN ORACLE INTEGRATION MESSAGE FOR ANY PROCESSING IN SUBSEQUENT ACTIONS (SEE SUPPORT FOR ADDING STANDARD AND CUSTOM SOAP AND HTTP HEADERS).

ENABLES YOU TO IMPLEMENT THE FOLLOWING MESSAGE EXCHANGE PATTERNS ON THE INBOUND SOAP ENDPOINT: SYNCHRONOUS REQUEST/RESPONSE, ONE-WAY REQUEST, AND ASYNCHRONOUS REQUEST WITH CALLBACK SUPPORT.

SUPPORTS TLS SERVER V1.2 IN THE TRIGGER (INBOUND) DIRECTION.

• SOAP ADAPTER CAPABILITIES WHEN CONFIGURED AS AN INVOCATION:

  ALLOWS INVOCATION OF AN HTTPS PROTOCOL-BASED EXTERNAL SOAP ENDPOINT, THEREBY ENCRYPTING THE COMMUNICATIONS USING TRANSPORT LAYER SECURITY (TLS) (SEE TRANSPORT LAYER SECURITY VERSION SUPPORT).

  ALLOWS INVOCATION OF HTTP PROTOCOL-BASED SOAP ENDPOINTS.

  ALLOWS INVOCATION OF EXTERNAL SOAP ENDPOINTS THAT ARE UNPROTECTED AND PROTECTED USING HTTP BASIC AUTHENTICATION AND WS-USERNAME TOKEN-BASED AUTHENTICATION.

  ALLOWS INVOCATION OF EXTERNAL SOAP ENDPOINTS HOSTED ON TLS SERVERS V1.1 AND V1.2.

  SUPPORTS INVOCATION OF TWO-WAY, SSL-ENABLED EXTERNAL SOAP ENDPOINTS (SEE TWO-WAY SSL SUPPORT FOR OUTBOUND CONNECTIONS).

  SUPPORTS CONFIGURATION OF STANDARD AND CUSTOM SOAP/HTTP HEADER PROPERTIES AVAILABLE TO THE OUTBOUND SOAP REQUEST (SEE SUPPORT FOR ADDING STANDARD AND CUSTOM SOAP AND HTTP HEADERS).


  SUPPORTS PROPAGATION OF THE SUBJECT BETWEEN COLocATED MODULES (FOR EXAMPLE, INTEGRATIONS TO PROCESSES AND PROCESSES TO INTEGRATIONS). THIS ENABLES THE MODULE TO PROVIDE CUSTOM FEATURES AND RESTRICTIONS BASED ON THE CURRENT SUBJECT. WHEN AN INTEGRATION INVOKES ANOTHER PROCESS OR INTEGRATION, THE SUBJECT IS PROPAGATED USING A JWT TOKEN. SIMILARLY, WHEN A PROCESS INVOKES AN INTEGRATION, IT PROPAGATES THE SUBJECT USING JWT (SEE SUPPORT FOR INVOKING COLocATED SOAP ENDPOINTS).
- Supports the dynamic discovery of endpoints. This is useful for scenarios in which the endpoint invoked by the SOAP Adapter must be dynamically configured based on runtime logic (see Support for Dynamic Endpoints).

- Supports the following:
  * Sending binary and nonbinary content as an MTOM attachment (up to 1 GB) as part of a request message while invoking external SOAP APIs.
  * Receiving binary and nonbinary content as an MTOM attachment (up to 1 GB) as part of a response message while invoking external SOAP APIs.

What Application Version Is Supported?

For information about which application version is supported by this adapter, see the Oracle Integration Adapters Certification Matrix under section Oracle Integration Adapters Certification at the top of the page:

Oracle Integration Adapters Certification Matrix

Workflow to Create and Add a SOAP Adapter Connection to an Integration

You follow a very simple workflow to create a connection with an adapter and include the connection in an integration in Oracle Integration.

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>More Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Create the adapter connections for the applications you want to integrate. The connections can be reused in multiple integrations and are typically created by the administrator.</td>
<td>Create a SOAP Adapter Connection</td>
</tr>
<tr>
<td>2</td>
<td>Create the integration. When you do this, you add trigger and invoke connections to the integration.</td>
<td>Create Integrations and Add the SOAP Adapter Connection to an Integration</td>
</tr>
<tr>
<td>3</td>
<td>Map data between the trigger connection data structure and the invoke connection data structure.</td>
<td>Map Data of Using Integrations in Oracle Integration</td>
</tr>
<tr>
<td>4</td>
<td>(Optional) Create lookups that map the different values used by those applications to identify the same type of object (such as gender codes or country codes).</td>
<td>Manage Lookups of Using Integrations in Oracle Integration</td>
</tr>
<tr>
<td>5</td>
<td>Activate the integration.</td>
<td>Manage Integrations of Using Integrations in Oracle Integration</td>
</tr>
<tr>
<td>6</td>
<td>Monitor the integration on the dashboard.</td>
<td>Monitor Integrations of Using Integrations in Oracle Integration</td>
</tr>
<tr>
<td>Step</td>
<td>Description</td>
<td>More Information</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>------------------</td>
</tr>
<tr>
<td>7</td>
<td>Track payload fields in messages during runtime.</td>
<td>Assign Business Identifiers for Tracking Fields in Messages and Manage Business Identifiers for Tracking Fields in Messages of Using Integrations in Oracle Integration</td>
</tr>
<tr>
<td>8</td>
<td>Manage errors at the integration level, connection level, or specific integration instance level.</td>
<td>Manage Errors of Using Integrations in Oracle Integration</td>
</tr>
</tbody>
</table>
The following sections describe SOAP Adapter capabilities in more detail.

Topics:

- SOAP Specifications
- Transport Layer Security Version Support
- Version Suppression of the Timestamp in the WS-Security Header
- Ability to Specify if the Timestamp is Not Required in the Response Message
- SOAP Action Validation Disabling for Inbound Requests
- Asynchronous Callback Response Support in the Invoke (Outbound) Direction
- Support for Adding Standard and Custom SOAP and HTTP Headers
- Support for Multiple Part Messages in Document-Style WSDLs
- Two-Way SSL Support for Outbound Connections
- SAML Policy Security Support in the Trigger (Inbound) Direction
- OAuth 2.0 Policy Security Support in the Trigger (Inbound) Direction
- Asynchronous Trigger Support in Orchestrated Integrations
- Support for Invoking Co-located SOAP Endpoints
- Support for Uploading a WSDL with Schemas in a ZIP File
- Support for Using MTOM to Transfer Large Binary Payloads
- Support for Dynamic Endpoints

SOAP Specifications

The following specifications are supported:

- SOAP 1.2
- WS-I Security (for SSL, TLS, and ciphers)
- SOAP 1.1 binding for MTOM
- WS-Addressing
- WS-Security Username Token

Transport Layer Security Version Support

Specifying the transport Layer Security (TLS) version of the target server is supported. The TLS protocol provides privacy and data integrity between two communicating computer applications. See Configure Connection Properties.
Version Suppression of the Timestamp in the WS-Security Header

Version suppression of the timestamp in the WS-Security header is supported. Suppression applies to the Username Password Token security policy in the invoke (outbound) direction. In secure Web Services transactions, a WS-Utility (WSU) timestamp can be inserted into a WS-Security header to define the lifetime of the message in which it is placed. See Configure Connection Properties.

Ability to Specify if the Timestamp is Not Required in the Response Message

You can specify if the timestamp is not required in the response message. See Configure Connection Properties.

SOAP Action Validation Disabling for Inbound Requests

Disabling SOAP action validation for inbound requests on the Operations page of the Adapter Endpoint Configuration Wizard is supported. This is useful for environments in which your WSDL includes custom code and you want to bypass validation. See Invoke Operation Page and Callback Integrations Fail with a Configured SOAP Action Mismatch Error.

Asynchronous Callback Response Support in the Invoke (Outbound) Direction

An asynchronous callback response in the invoke (outbound) direction is supported. This feature is enabled when the WSDL used in the connection defines a one-way operation in the selected service/port. The callback response endpoint must be specified through a different integration flow. The callback endpoint can be secured with any policy supported by the SOAP Adapter on the trigger side.

Based on the operation selection, if it is a one-way operation, you are asked to select the expected response type (no response or delayed response) on the Callback Operation page of the Adapter Endpoint Configuration Wizard.

- No Response: One-way invocation.
- Delayed Response: Specify the callback port type, operation, callback flow identifier, and version.

The callback flow identifier and version are used to determine the callback endpoint and sent in the ReplyTo header while sending a request to the outbound endpoint.

Support for Adding Standard and Custom SOAP and HTTP Headers

Adding standard and custom SOAP and HTTP headers to outbound and inbound requests and handling the responses with headers to propagate back to the user are supported. This configuration enables header configuration for the inbound service and header propagation for the outbound service. WS-Addressing headers propagation is not supported (for example, MessageId, ReplyTo, FaultTo, and so on). All header information and body elements are encapsulated under a single element so the mapper can display request and response information. See Add the SOAP Adapter Connection to an Integration.
Support for Multiple Part Messages in Document-Style WSDLs

Note:
The SOAP Adapter does not support RPC-style WSDL bindings. Only document-style WSDL bindings are supported.

- Multiple part messages in document-style WSDLs is supported. The support is provided for both inbound and outbound adapter configurations.

Standard SOAP headers can be defined in a WSDL in two ways:

- Implicit headers:
  With this type, the request header and body part are in different message types. In the binding section of the WSDL, the header uses the part name within the message type and message type name. The body does not have any part names explicitly defined in it.

```xml
<wSDL:message name="CreateUserRequestHeader">
  <wSDL:part name="requestHeader" element="tns:UserCreate"/>
</wSDL:message>
<wSDL:message name="CreateUserRequest">
  <wSDL:part element="tns:UserCreateHeader" name="parameters"/>
</wSDL:message>
<wSDL:binding name="UserBinding" type="tns:UserEndPoint">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wSDL:operation name="CreateUser">
    <soap:operation soapAction="http://example.com/CreateUser"/>
    <wSDL:input>
      <soap:body use="literal"/>
      <soap:header use="literal" part="requestHeader"
        message="tns:CreateUserRequestHeader"/>
    </wSDL:input>
    <wSDL:output>
      <soap:body use="literal"/>
    </wSDL:output>
  </wSDL:operation>
</wSDL:binding>
```

- Explicit headers:
  With this type, there are multiple parts in a single message type in the WSDL: one for the header and one for the body payload. The header is specified by its part name. The body uses its own name.

```xml
<wSDL:message name="CreateUserRequest">
  <wSDL:part element="tns:UserCreateHeader" name="parameters"/>
  <wSDL:part name="requestHeader" element="tns:UserCreate"/>
</wSDL:message>
<wSDL:binding name="UserBinding" type="tns:UserEndPoint">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wSDL:operation name="CreateUser">
    <soap:operation soapAction="http://example.com/CreateUser"/>
    <wSDL:input>
      <soap:body use="literal"/>
      <soap:header use="literal" part="requestHeader"
        message="tns:CreateUserRequestHeader"/>
    </wSDL:input>
    <wSDL:output>
      <soap:body use="literal"/>
    </wSDL:output>
  </wSDL:operation>
</wSDL:binding>
```
<soap:operation soapAction="http://example.com/CreateUser"/>
<wsdl:input> <soap:body use="literal"/>
<soap:header use="literal" part="requestHeader"
message="tns:CreateUserRequest"/>
</wsdl:input>
<wsdl:output>
<soap:body use="literal"/>
</wsdl:output>
</wsdl:operation>

Note:
Without specifying a header, multiple parts in a document-style WSDL are not supported.

When you invoke the Adapter Endpoint Configuration Wizard to configure the SOAP Adapter as a trigger or invoke, the Operations page detects that the WSDL includes defined SOAP request and/or response headers and automatically enables the button to configure SOAP headers for the endpoint. You can select No to remove the headers for the endpoint. You cannot modify these headers. The subsequent Request Header and Response Header pages of the WSDL load and show the specific headers defined in the WSDL. See Add the SOAP Adapter Connection to an Integration.

Two-Way SSL Support for Outbound Connections
The use of two-way SSL for outbound communications is supported. This feature enables an integration to invoke web services hosted on a two-way, SSL-enabled server and receive a response in return.

You must satisfy the following requirements to use this feature:

• Upload the following certificates in the Upload Certificate dialog in Oracle Integration. See Upload an SSL Certificate.
  – Upload a two-way SSL identity certificate type. This certificate is created from the client server on which two-way SSL must be enabled
  – Upload a trust certificate type for the outbound call. This is the certificate for the client server that hosts your web service.

• Specify a WSDL URL with secure HTTP (https) on the Connection Properties dialog. This WSDL must use the web service URL hosted on the two-way, SSL-enabled server.

• Two-way SSL is only supported with the JCA transport mechanism of the SOAP Adapter. The HTTP transport mechanism of the SOAP Adapter is not supported.

• Configure the server that hosts the web service for two-way SSL communication.

• Configure the SOAP Adapter as both a trigger and invoke. When you create the integration, you configure this same SOAP Adapter connection as both the trigger and invoke.
SAML Policy Security Support in the Trigger (Inbound) Direction

The Security Assertion Markup Language (SAML) is an XML-based, open-standard data format for exchanging authorization and authentication information between two different systems, typically an identity provider and a service provider. A SOAP Adapter trigger-specific connection can be configured to protect inbound SOAP endpoints using SAML token-based authentication. This configuration can be used to implement use cases that involve receiving callback messages from Oracle ERP Cloud upon completion of file-based data import (FBDI) jobs and upon completion of asynchronous operations in any Oracle ERP Cloud application, such as fusion order management (FOM). However, any client that supports SAML bearer token authentication can use this policy.

OAuth 2.0 Policy Security Support in the Trigger (Inbound) Direction

Integrations exposing SOAP endpoints using the SOAP Adapter as a trigger connection can be OAuth 2.0-protected. OAuth 2.0 is an industry-standard protocol for authorization. OAuth 2.0 focuses on client developer simplicity while providing specific authorization flows for web applications.

Asynchronous Trigger Support in Orchestrated Integrations

When a call is made to an asynchronous service, the response is expected at a later time and possibly to a different endpoint. If the response is expected at a different endpoint, the endpoint information must be passed to the asynchronous service during the request using the WS-Addressing ReplyTo header. In this case, the Oracle Integration endpoint with the SOAP Adapter configured as the (trigger) inbound connection acts as an asynchronous service. Oracle Integration determines if the selected operation is asynchronous and then enables you to provide callback endpoint details through a ReplyTo standard header and to use this information to invoke the callback response.

The following requirements must be satisfied to use this feature:

- This feature is only available when the SOAP Adapter is included in orchestrated integrations.
- The asynchronous service uses only the SOAP Adapter-supported OWSM policies. The callback endpoint being specified in the ReplyTo header must support one of the security policies available on the Connections page for the invoke-only role.
- The corresponding callback invoke must also be configured when the trigger is configured for an asynchronous response.
- The request payload contains a Reply-To header that contains the value of the endpoint to which to send the asynchronous response.

In the trigger direction, you must configure the Adapter Endpoint Configuration Wizard as follows:

- On the Callback Operations Page, you select **Delayed Response** because a callback response is expected.
- On the Headers page, the **Do you want to configure headers for this Endpoint** option is automatically enabled. The **SOAP Headers** option is automatically selected in the **Request Headers** section and cannot be changed.
On the Request Headers page, the WS-Addressing ReplyTo, MessageID, and Action headers are automatically populated in the Standard SOAP Headers tab.

In the invoke direction, you must configure the Adapter Endpoint Configuration Wizard as follows:

- On the Welcome page, select Yes to configure the SOAP Adapter as a callback invoke.
- On the Operations page, the list of port types is displayed (instead of the service and port). You must select the callback port type and callback operation. The Callback Operations page is disabled as it is not an outbound asynchronous case.
- On the Headers page, the headers configuration option is automatically enabled. The SOAP Headers option is automatically selected in the Request Headers section and cannot be changed.
- On the Request Headers page, the WS-Addressing ReplyTo, MessageID, Action headers are populated in the Standard SOAP Headers tab.

During integration creation, you must explicitly map the WS-Addressing headers from Inbound Request to Callback Invoke Request in addition to the other required mapping options. During runtime, when Oracle Integration is invoked with the request having a wsa:ReplyTo header, the service invokes the endpoint sent in the header with the response.

**Note:** You cannot switch from an asynchronous trigger/callback invoke to nonasynchronous trigger/invoke.

### Support for Invoking Co-located SOAP Endpoints

You can propagate the subject between co-located modules (for example, Integrations to Processes and Processes to Integrations). This enables the module to provide custom features and restrictions based on the current subject.

Oracle Integration can automatically determine if an outbound (invoke) SOAP endpoint being invoked by an integration is local (co-located) or remote to Oracle Integration and then optimize the invoke call to the endpoint. Co-located means the integrations are running on the same host instance or in the same domain. If the outbound endpoint is co-located, the endpoint is invoked using an optimized HTTP request using a JSON Web Token (JWT) token for authorization. The optimized HTTP request is a plain HTTP request (non-SSL) sent directly to the managed server. The currently configured security policy is overwritten by a JWT token. JWT is a JSON-based open standard (RFC 7519) for creating access tokens that assert some number of claims. For example, a server can generate a token that has the claim "logged in as admin" and provide that to a client. The client can then use that token to prove that it is logged in as admin. The tokens are signed by the server's key. Therefore, the client and server can both verify that the token is legitimate.

### Support for Uploading a WSDL with Schemas in a ZIP File

When creating a connection, you can upload a ZIP file with a WSDL and dependencies such as other WSDLs and XSDs nested inside the ZIP. This can be useful in scenarios in which you adopt the standard canonical model (OAGIS) in integrations. If you upload a ZIP, you must ensure that the WSDLs are available in the
top two directory levels. This makes the WSDLs available for selection in the Service WSDL dropdown list on the Connections page. There can be any number of WSDLs at these two levels. Any WSDLs below these levels do not appear for you to select in the Service WSDL dropdown list.

The following use cases are supported:

- The ZIP file contains the main WSDL (for this example, named SalesOrderEBSV2.wsdl) in the immediate folder and its dependencies (EBM, other XSDs, and so on) nested deeply inside a folder.

For this example, the main WSDL reference must be corrected to reference EnterpriseObjectLibrary as shown.

```xml
<xsd:import namespace="http://xmlns.oracle.com/EnterpriseObjects/Core/Common/V2" schemaLocation="EnterpriseObjectLibrary/Common/V2/Meta.xsd"/>
```

- The ZIP file contains a main WSDL (for this example, named service.wsdl) and references another WSDL and XSDs:

- The ZIP file contains a WSDL (for this example, named service.wsdl) and any number of XSD dependencies in the same directory.
Support for Using MTOM to Transfer Large Binary Payloads

Message Transmission Optimization Mechanism (MTOM) support is provided. MTOM is a W3C Message Transmission Optimization Mechanism, a method for efficiently sending binary data to and from web services. Binary objects in SOAP are represented as base64 encoded messages, which essentially expands the data by about 33%. For large payloads, this can significantly impact performance and transmission time. MTOM provides a solution to transfer a large binary payload using optimization. MTOM/XOP optimizes a SOAP message and the XOP processing serializes it into a MIME multipart/related message. The XOP processing extracts the base64-encoded data from the SOAP message and packages it as separate binary attachments within the MIME message. See Configure MTOM Support in the SOAP Adapter.

MTOM is currently supported only on the invoke connection in an integration.

Note:

MTOM upload/download cannot be invoked asynchronously with a large payload and high concurrency. This scenario can result in an out-of-memory error depending upon payload size and concurrency. Take care in your design. For example, in a scheduled orchestrated integration with scheduling set to every 10 minutes, four flows can run consistently with 512 MB payload every 10 minutes on a two-node Oracle Integration cluster without any out-of-memory errors.

Support for Dynamic Endpoints

The SOAP Adapter supports the dynamic discovery of endpoints. This is useful for scenarios in which the endpoint that the SOAP Adapter invokes must be dynamically configured based on runtime logic. This feature is applicable for both new integrations and existing integrations (edited to add new invokes) that include the SOAP Adapter as an invoke connection. The endpoint invoked must support the same security policies as supported in Oracle Integration. However, WS-Addressing is not used. Instead, two types of properties in the mapper are provided for configuring dynamic invocation. These properties are used during runtime to override the properties configured on the Connections page during design time.

- Endpoint Properties: Override the endpoint details.
  - EndpointURI: Replaces the existing endpoint URI specified on the Connections page before invoking the endpoint.
  - SoapAction: Replaces the existing SOAP action validation setting before invoking the endpoint.

- Security Properties: Override the endpoint credential details, if required.
  - If the connection uses Username Token Security:
    * Username: Replaces the username credentials before invoking the endpoint.
    * Password: Replaces the password credentials before invoking the endpoint.
  - If the connection uses Basic Authentication Security:
Authorization: Replaces the authorization header before invoking the endpoint.

The following dynamic properties are visible for configuration in the mapper and used during runtime to override the properties configured during design time.

- Sample outbound request: The **ConnectivityProperties** section and the **Headers** and/or **Body** sections are displayed with properties for configuration. The **ConnectivityProperties** section is not visible if this is a callback invoke request.

- Username Token security policy: The **EndpointProperties** section and **SecurityProperties** section are visible with properties for configuration under the **ConnectivityProperties** section.
Basic Authentication security policy: The **EndpointProperties** section and **SecurityProperties** section are visible with properties for configuration under the **ConnectivityProperties** section.
- No security policy: The **EndpointProperties** section is visible with properties for configuration under the **ConnectivityProperties** section.
Note:

- Overriding security properties with the dynamic endpoint invocation feature logs the details mapped when trace is enabled during activation of the integration.

- When using dynamic endpoints with the SOAP Adapter, be aware that if you activate an integration using this feature with Enable tracing and Include payload selected, the password used in the payload during runtime is exposed in clear text in the log file.
Create a SOAP Adapter Connection

A connection is based on an adapter. You define connections to the specific cloud applications that you want to integrate.

Topics:

• Prerequisites for Creating a Connection
• Create a Connection
• Upload an SSL Certificate

Prerequisites for Creating a Connection

Ensure that the WSDL to use is reachable. There is no restriction on the type of WSDL to use.

To create an integration that consumes external REST APIs hosted on a two-way, SSL-enabled server, satisfy the following prerequisites:

• Ensure that the server on which the external REST APIs are hosted is enabled for two-way SSL support.
• File a service request with Oracle Support Services to obtain the keystore file required for establishing an Oracle Integration identity to facilitate a two-way SSL-based integration.
• Import the necessary keystore file and trust certificate.

Create a Connection

The first step in creating an integration is to create the connections to the applications with which you want to share data.

1. In the navigation pane, click Integrations, then click Connections.
2. Click Create.

Note:

You can also create a connection in the integration canvas of:

• An orchestrated integration (See Define Inbound Triggers and Outbound Invokes.)
• A basic routing integration (See Add a Trigger (Source) Connection.)
The Create Connection — Select Adapter dialog is displayed.

3. Select an adapter from the dialog. You can also search for the type of adapter to use by entering a partial or full name in the Search field, and clicking Search. The Create New Connection dialog is displayed.

4. Enter the information to describe the connection.
   - Enter a meaningful name to help others find your connection when they begin to create their own integrations. The name you enter is automatically added in capital letters to the Identifier field. If you modify the identifier name, do not include a blank space (for example, Sales Opportunity).
   - Select the role (direction) in which to use this connection (trigger, invoke, or both). Only the roles supported by this adapter are displayed for selection. When you select a role, only the connection properties and security policies appropriate to that role are displayed on the Connections page. If you select an adapter that supports both invoke and trigger, but select only one of those roles, then try to drag the adapter into the section you did not select, you receive an error (for example, configure an Oracle Service Cloud (RightNow) Adapter as only an invoke, but drag the adapter to the trigger section).
   - Enter an optional description of the connection.

5. Click Create.

Your connection is created and you are now ready to configure connection details, such as email contact, connection properties, security policies, connection login credentials, and (for certain connections) agent group.

Add a Contact Email

From the Connection Administrator section of the connection, you can add a contact email address for notifications.
1. In the **Email Address** field, enter an email address to receive email notifications when problems occur.

2. In the upper right corner, click **Save**.

**Configure Connection Properties**

Enter connection information so your application can process requests.

1. Click **Configure Connectivity**.
   The Connection Properties dialog is displayed.

2. In the **WSDL URL** field, specify the URL in either of two ways:
   a. Click the **Upload File** checkbox, then click **Upload** to upload the WSDL.
      If you upload a ZIP file, the file is validated and the page is refreshed to display the **Service WSDL** list. The relative paths of all WSDLs in the ZIP are displayed. Select the WSDL to use in the connection.
   b. Manually specify the WSDL to use.

3. In the **Target Server's TLS version (Optional)** field, optionally specify the Transport Layer Security (TLS) version of the target server. Oracle Integration is configured to use TLS v1.1 by default.
   - TLSv1.1
   - TLSv1.2

   The TLS protocol provides privacy and data integrity between two communicating computer applications. Depending on the JDK version in use, the version of TLS being used during connection establishment can differ for different clients and servers. If the outbound server requires a different TLS version, specify the required version to use.

4. In the **Suppress insertion of timestamp into the request (Optional)** field, optionally suppress the timestamp in the WS-Security header. Suppression applies to the Username Password Token security policy in the invoke (outbound) direction. In secure Web Services transactions, a WS-Utility (WSU) timestamp can be inserted into a WS-Security header to define the lifetime of the message in which it is placed.
   - **Yes**: No timestamp is added to the WS-Security header sent as part of the outbound request. For inbound requests with the basic authentication security policy, no timestamp is required to be sent by the client.
   - **No**: Clients are expected to send a timestamp in the WS-Security header with the request.
5. In the **Ignore timestamp in the response message (Optional)** field, specify if the timestamp is not required in the response message.
   - **Yes**: The timestamp is not required in the response message. If the timestamp is present in the SOAP security header when the response is received from the service, it is ignored.
   - **No**: The timestamp is received in the response from the service is not ignored.

6. In the **Enable two way SSL for outbound connection** field, select **Yes** if you want to enable two-way SSL communication.

7. In the **Name of the certificate to use for SSL communication** field, enter the key alias name configured for two-way SSL communication. This is the name you entered in the **Key Alias Name** field when uploading the identity certificate in the Upload Certificate dialog. See **Upload an SSL Certificate**.

8. Click **OK**.


**Configure Connection Security**

Configure security for your Update Adapter Product Name connection by selecting the security policy.

1. Click **Configure Credentials**.

2. Select the security policy.
   
The page is refreshed to display the login credential fields.

3. Specify the login credentials. For trigger (inbound) connections, the security policy must be either username password token, basic authentication, SAML, or OAuth 2.0. This is because all Oracle Integration inbound endpoints are protected with either of these policies.
Security Policy | Fields
--- | ---
**Basic Authentication**
(In the trigger (inbound) direction, supports HTTP basic authentication over SSL: oracle/wss_http_token_over_ssl_service_policy).

Note the following behavior:

- If the invoking client is secured with Oracle Web Services Manager (OWSM) using an oracle/wss* policy, the client receives a failure.
- In the inbound (trigger) direction, if the **Suppress insertion of timestamp into the request (Optional)** field is enabled, then oracle/http_basic_auth_over_ssl_service_policy is supported.
- In customer-managed environments, when configuring a trigger SOAP Adapter with Basic Authentication, the wss_http_token_service_policy policy is used regardless of whether the **Suppress insertion of timestamp into the request** option is set to **Yes** or **No** in the Connections page. Therefore, with or without the timestamp added in the SOAP header, as long as the username and password credentials are valid, the connection runs successfully at runtime.

If Basic Authentication is required for both trigger and invoke connections, create one connection with the **Trigger and Invoke** role that uses the Basic Authentication security policy.

**Username Password Token**
(In the trigger (inbound) direction, supports oracle/wss_username_token_over_ssl_service_policy.)

- **Username** — Enter the name of a user who has access to the destination web service.
- **Password** — Enter the password.
- **Confirm Password** — Reenter the password.

---

<table>
<thead>
<tr>
<th>Security Policy</th>
<th>Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Basic Authentication</strong></td>
<td><strong>Username</strong> — Enter the name of a user who has access to the destination web service. <strong>Password</strong> — Enter the password. <strong>Confirm Password</strong> — Reenter the password.</td>
</tr>
</tbody>
</table>

---
### Security Policy

<table>
<thead>
<tr>
<th>Security Policy</th>
<th>Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Security Policy</td>
<td>No fields are displayed.</td>
</tr>
</tbody>
</table>
| **Security Assertion Markup Language (SAML)** | This policy is *only* available when configuring the SOAP Adapter as a trigger. If you attempt to add the SOAP Adapter with this security policy configuration as an invoke in an integration, you receive an error.  
  - **Username** — Optionally enter the name of the SAML user. |
| OAuth 2.0 | This policy is *only* available when configuring the SOAP Adapter as a trigger. If you attempt to add the SOAP Adapter with this security policy configuration as an invoke in an integration, you receive an error.  
  No fields are displayed. |

If you select a security policy, the following behavior occurs.

<table>
<thead>
<tr>
<th>If the Inbound SOAP Connection is Configured with Security Policy...</th>
<th>Then...</th>
</tr>
</thead>
</table>
| **Username Password Token** | • The client should send the username/password and timestamp as part of the WSEE header.  
  • The response includes only the SOAP payload. |
| **Basic Authentication** | • The client should send the username/password in the HTTP headers and timestamp as part of the WSEE header.  
  • The response includes only the SOAP payload. |
| **Basic Authentication and the Suppress insertion of timestamp into the request (Optional) field is enabled** | • The client should send the username/password in the HTTP headers.  
  • The response includes only the SOAP payload. |

**Note:**

If no timestamp is included as part of the header, configure the SOAP Adapter connection with the Basic Authentication security policy (oracle/http_basic_auth_over_ssl_client_policy) and set **Suppress insertion of timestamp into the request (Optional)** to Yes.

4. Click **OK**.

### Configure an Agent Group

Configure an agent group for accessing the service hosted on your premises behind the fire wall.

1. Click **Configure Agents**.

The Select an Agent Group page appears.

2. Click the name of the agent group.

3. Click **Use**.
To configure an agent group, you must download and install the on-premises connectivity agent. See Download and Run the On-Premises Agent Installer and About Agents and Integrations Between On-Premises Applications and Oracle Integration in Using Integrations in Oracle Integration.

Test the Connection

Test your connection to ensure that it is successfully configured.

1. In the upper right corner of the page, click **Test**.
2. If your adapter connection uses a WSDL, you are prompted to select the type of connection testing to perform:
   - **Validate and Test**: Performs a full validation of the WSDL, including processing of the imported schemas and WSDLs. Complete validation can take several minutes depending on the number of imported schemas and WSDLs. No requests are sent to the operations exposed in the WSDL.
   - **Test**: Connects to the WSDL URL and performs a syntax check on the WSDL. No requests are sent to the operations exposed in the WSDL.

   If successful, the following message is displayed and the progress indicator shows 100%.
   
   Connection **connection_name** was tested successfully.

3. If your connection was unsuccessful, an error message is displayed with details. Verify that the configuration details you entered are correct.
4. When complete, click **Save**, then click **Close**.

Upload an SSL Certificate

Certificates are used to validate outbound SSL connections. If you make an SSL connection in which the root certificate does not exist in Oracle Integration, an exception is thrown. In that case, you must upload the appropriate certificate. A certificate enables Oracle Integration to connect with external services. If the external endpoint requires a specific certificate, request the certificate and then upload it into Oracle Integration.

To upload an SSL certificate:

1. In the navigation pane, click **Integrations**, then click the < arrow next to **Designer**.
2. Click **Settings > Certificates**.

   All certificates currently uploaded to the trust store are displayed in the Certificates dialog. The **Filter By > Type** list displays the following details:
   - **Preinstalled**: Displays the certificates automatically installed in Oracle Integration. These certificates cannot be deleted.
   - **Uploaded**: Displays the certificates uploaded by individual users. These certificates can be deleted and updated.

   You can also search for certificates in the **Search** field. The search results are limited to a maximum of ten records sorted by name for performance and usability reasons. To ensure that your search results are more granular, enter as much of the certificate name as possible.

3. Click **Upload** at the top of the page.
4. In the Upload Certificate dialog box, select the certificate type. Each certificate type enables Oracle Integration to connect with external services.

   - **Trust Certificate**: Use this option to upload a trust certificate.
     a. Enter a unique alias for the certificate.
     b. Click Browse, then select the trust file (for example, .cer or .crt) to upload.

   - **Message Protection Certificate**: Use this option to upload a keystore certificate with SAML token support. Create, read, update, and delete (CRUD) operations are supported on this type of certificate.
     a. Enter a unique alias for the certificate.
     b. Click Browse, then select the certificate file (.cer or .crt) to upload.

   - **Identity Certificate**: Use this option to upload a certificate for two-way SSL communication.
     a. Click Browse, then select the keystore file (.jks) to upload.
     b. Enter the password of the keystore being imported.
     c. Enter the comma-separated list of aliases from the keystore being imported.
     d. Enter the comma-separated list of passwords corresponding to key aliases.
     e. If you want to display the passwords in clear text, select **Show Key Password(s)**. This enables you to ensure that you are correctly entering a list of keystore passwords.

5. Click **Upload**.

6. Click the certificate name to view details such as the subject of the certificate, the issuer of the certificate, the date the certificate was issued, and the date the certificate expires.
Add the SOAP Adapter Connection to an Integration

When you drag the SOAP Adapter into the trigger or invoke area of an integration, the Adapter Endpoint Configuration Wizard appears. This wizard guides you through the configuration of the SOAP Adapter endpoint properties.

These topics describe the wizard pages that guide you through configuration of the SOAP Adapter as a trigger or invoke in an integration.

Topics:
- Basic Info Page
- Trigger Operation Page
- Trigger Callback Operation Page
- Invoke Operation Page
- Header Page
- Request Header Page
- Response Header Page
- Summary Page

Basic Info Page

You can enter a name and description on the Basic Info page of each adapter in your integration.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>What do you want to call your endpoint?</td>
<td>Provide a meaningful name so that others can understand the responsibilities of this connection. You can include English alphabetic characters, numbers, underscores, and dashes in the name. You cannot include the following:</td>
</tr>
<tr>
<td></td>
<td>• Blank spaces (for example, My Inbound Connection)</td>
</tr>
<tr>
<td></td>
<td>• Special characters (for example, #;83 or righ(t)now4)</td>
</tr>
<tr>
<td></td>
<td>• Multibyte characters</td>
</tr>
<tr>
<td>What does this endpoint do?</td>
<td>Enter an optional description of the connection’s responsibilities. For example: This connection receives an inbound request to synchronize account information with the cloud application.</td>
</tr>
</tbody>
</table>
Do you want to configure this as a callback invoke?

Select Yes to configure the SOAP Adapter as a callback invoke. This option is only available when configuring the SOAP Adapter as an invoke in an orchestrated integration. See Asynchronous Trigger Support in Orchestrated Integrations.

### Trigger Operation Page

Enter the port type and operation for the SOAP Adapter. If your WSDL includes only a single service, port type, and operation, they are automatically selected. If the WSDL includes multiple services and port types, then select the ones to use in your integration. Based on the selected values, other objects such as the request object, response object, and fault object may also be automatically displayed.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selected Port Type</td>
<td>Displays the selected port type. If your WSDL includes multiple port types, select the port type.</td>
</tr>
<tr>
<td>Select the Operation</td>
<td>Displays the selected operation. If your WSDL includes multiple operations, select the operation.</td>
</tr>
<tr>
<td>Request Object</td>
<td>Displays the request object (if your WSDL includes request objects).</td>
</tr>
<tr>
<td>Response Object</td>
<td>Displays the response object (if your WSDL includes response objects).</td>
</tr>
<tr>
<td>Disable SoapAction validation</td>
<td>Select Yes to disable SOAP action validation for inbound requests. This is useful for environments in which your WSDL includes custom code and you want to bypass validation. When set to No (the default), Oracle Integration validates the SOAP action to ensure that it matches the WSDL.</td>
</tr>
</tbody>
</table>

### Trigger Callback Operation Page

Enter the trigger callback response operation details for the SOAP Adapter. This page is only displayed when the SOAP Adapter is used in orchestrated integrations or a one-way operation is selected on the Operations page.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Response</td>
<td>Select if a one-way call without a response is expected.</td>
</tr>
<tr>
<td>Delayed Response</td>
<td>Select if a delayed callback response is expected.</td>
</tr>
</tbody>
</table>
Invoke Operation Page

Enter the service, port, and operation for the SOAP Adapter invoke connection or enter the port type and operation for the SOAP Adapter callback invoke connection. If the WSDL file you specified during SOAP Adapter connectivity configuration includes only a single service, port type, or operation, they are automatically selected for use. If the WSDL included multiple services, port types, or operations, then select the ones to use in this integration.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selected Port Type</td>
<td>Displays the selected port type. If your WSDL includes multiple port types, select the port type.</td>
</tr>
<tr>
<td>Selected Operation</td>
<td>Displays the selected operation. If your WSDL includes multiple operations, select the operation.</td>
</tr>
<tr>
<td>Request Object</td>
<td>Displays the request object (if your WSDL includes request objects).</td>
</tr>
<tr>
<td>Response Object</td>
<td>Displays the response object (if your WSDL includes response objects).</td>
</tr>
</tbody>
</table>

Header Page

Enter the header details for the SOAP Adapter. The following table describes the key information on the Oracle SOAP Adapter Header page. The headers you specify are applied to the request and/or response object of the selected operation. The selected elements are included under respective wrapper elements in the integration WSDL and are displayed in the mapper as a request and/or response.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure MTOM Attachment Options</td>
<td>MTOM attachment options are shown when a base64Binary element is present in the WSDL messages for a given operation in both the request and response messages (for synchronous integrations).</td>
</tr>
<tr>
<td></td>
<td>• <strong>Send attachments in request</strong>: Select to configure MTOM for the outbound request.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Accept attachments in response</strong>: Select to configure MTOM for the outbound response.</td>
</tr>
</tbody>
</table>
### Element Description

| Element地上 want to configure headers for this Endpoint? | Select Yes, then select the headers to include. Yes is automatically selected for you in the following situations:  
- Your endpoint already contains SOAP headers.  
- An asynchronous trigger (trigger with a one-way operation selected on the Operations Page and Delayed Response selected on the Callback Operations page) or callback invoke (invoke with Do you want to configure this as a callback invoke? set to Yes on the Basic Info page) was configured. The selection is disabled in this case and cannot be modified. |
| --- | --- |
| SOAP Headers | This option is automatically selected in the following situations and cannot be modified:  
- Your endpoint already contains SOAP headers.  
- An asynchronous trigger (trigger with a one-way operation selected on the Operations Page and Delayed Response selected on the Callback Operations page) or callback invoke (invoke with Do you want to configure this as a callback invoke? set to Yes on the Basic Info page) was configured. |
| Standard HTTP Headers | Select this check box to add standard HTTP headers in the request and/or response. |
| Custom HTTP Headers | Select this check box to add custom HTTP headers in the request and/or response. |
| Custom SOAP Headers | Select this check box to add custom SOAP headers in the request and/or response. |

**Note:**

Based on the selections made on this page, separate tabbed pages are shown in the Request Header page and/or Response Header page for configuring the selected headers.

---

### Request Header Page

Enter the request header details for the SOAP Adapter. You can configure and view standard HTTP, custom HTTP, and custom SOAP request header parameters for the SOAP Adapter based on the selections made on the Header page.
### Response Header Page

Enter the response header details for the SOAP Adapter. You can configure and view standard HTTP, custom HTTP, and custom SOAP response header parameters for the SOAP Adapter based on the selections made in the Header page.

<table>
<thead>
<tr>
<th>Tabbed Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOAP Headers</td>
<td>View the SOAP headers contained in the WSDL defined on the Connections page. These headers cannot be modified.</td>
</tr>
<tr>
<td>Standard HTTP Headers</td>
<td>Configure the standard HTTP headers. Click the Add icon to add headers from the prepopulated list. Some of the mandatory standard HTTP headers are disabled for selection because allowing them to change may provide unexpected results (for example, authorization).</td>
</tr>
<tr>
<td>Custom HTTP Headers</td>
<td>Configure the custom HTTP headers. Click the Add icon to add custom HTTP header names and descriptions.</td>
</tr>
<tr>
<td>Custom SOAP Headers</td>
<td>Configure the custom SOAP headers. Click Browse to add a schema file from which to select custom SOAP headers to be configured in the integration WSDL. Select a header element from the uploaded schema and click Add Header. See Configure Custom SOAP Headers for the SOAP Adapter.</td>
</tr>
</tbody>
</table>
### Tabbed Page

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Custom SOAP Headers</td>
</tr>
<tr>
<td>Configure the custom SOAP headers. Click <strong>Browse</strong> to add a schema file from which to select custom SOAP headers to be configured in the integration WSDL. Select a header element from the uploaded schema and click <strong>Add Header</strong>. See <a href="#">Configure Custom SOAP Headers for the SOAP Adapter</a>.</td>
</tr>
</tbody>
</table>

---

## Invoke Callback Operation Page

Enter the callback response operation details for the SOAP Adapter. This page is also visible for the normal invokes (*Do you want to configure this as a callback invoke* is set to **No** on the Basic Info Page) case in which the one-way operation was selected on the Operations Page.

### Element Description

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Response</td>
<td>Select if a no callback response is expected.</td>
</tr>
<tr>
<td>Delayed Response</td>
<td>Select if a delayed callback response is expected.</td>
</tr>
<tr>
<td>Select the Port Type</td>
<td>Select the port type to use for the asynchronous callback response.</td>
</tr>
<tr>
<td>Selected Callback Operation</td>
<td>View the callback operation associated with the selected port type. In the case of multiple operations, Select the operation.</td>
</tr>
<tr>
<td>Flow Identifier</td>
<td>Specify the name of the callback integration. (that is, to be used in the request integration). The identifier value must be the same as the callback integration flow identifier seen in Oracle Integration integrations.</td>
</tr>
<tr>
<td>Flow Version</td>
<td>Specify the version number of the callback integration. The version value must be the same as the callback integration flow version seen in Oracle Integration integrations.</td>
</tr>
</tbody>
</table>
You can review the specified adapter configuration values on the Summary page.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>Displays a summary of the configuration values you defined on previous pages of the wizard. The information that is displayed can vary by adapter. For some adapters, the selected business objects and operation name are displayed. For adapters for which a generated XSD file is provided, click the XSD link to view a read-only version of the file. To return to a previous page to update any values, click the appropriate tab in the left panel or click Back. Click Cancel to cancel your configuration details.</td>
</tr>
</tbody>
</table>
Implement Common Patterns Using the SOAP Adapter

You can use the SOAP Adapter to implement the following common patterns.

Topics:
• Configure MTOM Support in the SOAP Adapter
• Consume Taleo SOAP APIs
• Invoke a SOAP-Based Integration with a Timestamp
• Configure Custom SOAP Headers for the SOAP Adapter
• Call an Oracle Fusion Applications Business Intelligence Publisher Report Synchronously

Configure MTOM Support in the SOAP Adapter

This use case describes how to configure Message Transmission Optimization Mechanism (MTOM) support in the SOAP Adapter.

See SOAP Adapter Capabilities for conceptual information.

SOAP Message Examples and Structure

The following example shows a SOAP message with inline binary content:

MIME-Version: 1.0
Content-Type: Multipart/Related; boundary=MIME_boundary;
type="application/soap+xml"; start="<claim@insurance.com>"

--MIME_boundary
Content-Type: application/soap+xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
Content-ID: <claim@insurance.com>

<Envelope>
  <Body>
    <ReceiveImage>
      <filename>abc.jpg</filename>
      <image>.... JPEG image base64 .....</image>
    </ReceiveImage>
  </Body>
</Envelope>
The following example shows a SOAP message with MTOM/XOP:

```xml
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xop='http://www.w3.org/2004/08/xop/include'
xmlns:xop-mime='http://www.w3.org/2005/05/xmlmime'>
  <soap:Body>
    <Order>
      <orderNumber>ABC</orderNumber>
      <orderType>backorder</orderType>
      <image xop-mime:content-type='image/jpeg'>
        <image xop-mime:content-type='image/jpeg'>
          ...
        </image>
      </image>
    </Order>
  </soap:Body>
</soap:Envelope>
```

--MIME_boundary

Content-Type: image/jpeg
Content-Transfer-Encoding: binary
Content-ID: <image@insurance.com> 4

...binary JPG image...

--MIME_boundary--

The MTOM message structure is as follows:

- The start parameter indicates which part of the MIME message is the root XOP document.
- The Content-ID value identifies a part of the MIME message. In this case, it is the root XOP document.
- The <xop:Include> element references the JPEG binary attachment.
- The Content-ID identifies the JPEG in the binary attachment.

**application/octet-stream MIME attachment**

A MIME attachment with the content type application/octet-stream is a binary file. It is typically an application or document that must be opened in an application, such as a spreadsheet or word processor. If the attachment has a file name extension associated with it, you may be able to identify the type of file. For example, an .exe extension indicates it is a Windows or DOS program (executable). A file ending in .doc can probably be opened in Microsoft Word. In addition to the generic application/octet-stream content type, you may also encounter attachments that have different subtypes (for example, application/postscript, application/x-macbinary, and application-msword). They are similar to application/octet-stream, but apply to specific types of files.

SOAP Message Transmission Optimization Mechanism/XML-binary Optimized Packaging (MTOM/XOP) describes a method for optimizing the transmission of XML
data of type `xs:base64Binary` in SOAP messages. When the transport protocol is HTTP, MIME attachments carry that data while at the same time allowing both the sender and the receiver direct access to the XML data in the SOAP message without having to be aware that any MIME artifacts were used to marshal the `xs:base64Binary` data. The binary data optimization process involves the following:

- Encoding the binary data
- Removing the binary data from the SOAP envelope
- Compressing and attaching the binary data to the MIME package
- Adding references to that package in the SOAP envelope

When MTOM is enabled, the MTOM specification does not require that the web service runtime use XOP binary optimization when transmitting base64binary data. Instead, the specification enables runtime to choose to do so. This is because in certain cases the runtime may decide that it is more efficient to send base64binary data directly in the SOAP message (for example, when transporting small amounts of data in which the overhead of conversion and transport consumes more resources than just inlining the data as is). However, the Oracle WebLogic Server web services implementation for the MTOM for JAX-RPC service always uses MTOM/XOP when MTOM is enabled.

**Design Time**

When you configure the SOAP Adapter as an invoke connection in an integration, MTOM attachment options are shown in the Adapter Endpoint Configuration Wizard when a base64Binary element is present in the WSDL messages for a given operation in both request and response messages (for synchronous).

1. Specify the base64Binary element-based WSDL in the Connections page when configuring the SOAP Adapter.
2. Enable the appropriate **Send attachments in request** (for outbound request) and **Accept attachments in response** (for outbound response) options to enable MTOM processing for that endpoint.

MTOM support cannot be configured in a trigger connection.

If you select to enable MTOM for a request message, the XPath for that binary node is calculated and stored in the JCA file as an interaction spec property.

For example:

```xml
<property name="attachmentXpathInfo" value="//*[namespace-uri()='http://www.oracle.com/UCM' and local-name()='GenericRequest']/*[namespace-uri()='http://www.oracle.com/'..."
```
For a response message, the property `attachmentXpathInfoForResponse` is used in the JCA file to represent the XPath.

Mapping

1. For an outbound request, any attachment reference from the virtual file system (VFS) can be mapped to the base64Binary element of the outbound message. As shown below, `attachmentReference` from the REST source is mapped to the base64Binary element of the message.

2. For an outbound response, the attachment is saved to the VFS and the base64Binary element of the payload holds the VFS reference. The VFS reference can be further used to map it to a REST resource or an FTP Adapter:
Runtime

During runtime, MTOM processing is triggered based on the availability of the `attachmentXpathInfo` and `attachmentXpathInfoForResponse` properties in the JCA file. This information is persisted during design time.

In the outbound request, the XPath information given by `attachmentXpathInfo` in the JCA file creates an attachment in the cloud message and structures the SOAP message in the MTOM-specific format.

In the outbound response, the logic checks if there is any attachment received in the response of the cloud message, which is further saved in the VFS. The node represented by the property `attachmentXpathInfoForResponse` substitutes it with the VFS reference of the attachment in the cloud message.

Consume Taleo SOAP APIs

Taleo enterprise edition SOAP APIs are protected. To use these APIs with the SOAP Adapter, you must use the WSDL file upload feature in the Connection Properties dialog of the Connections page instead of specifying the WSDL with the HTTP URI.

See Configure Connection Properties.

Invoke a SOAP-Based Integration with a Timestamp

You can use an interface such as the SOAP UI to invoke a SOAP-based web service integration in Oracle Integration. If you attempt to invoke a SOAP-based web service through the SOAP UI, you can receive the following error regardless of the security policy you selected in the Connections page (Username Password Token, Basic Authentication, or No Security Policy).

OWSM ICS Service request handler failed: InvalidSecurity

If this error occurs, add a timestamp to your WSS security. For example:

```xml
<soap:Header>
  <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
    <wsu:Timestamp wsu:Id="IS-918F6052E6918D5F2414550387589204">
      <wsu:Created>2017-04-17T17:20:50.9200</wsu:Created>
      <wsu:Expires>2017-04-17T17:21:50.9200</wsu:Expires>
    </wsu:Timestamp>
    <wsse:UsernameToken>
      <wsse:Username>Joe</wsse:Username>
      <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText">My_password</wsse:Password>
    </wsse:UsernameToken>
  </wsse:Security>
</soap:Header>
```
Configure Custom SOAP Headers for the SOAP Adapter

Oracle Integration supports adding custom SOAP headers that are not defined in the WSDL definition provided on the Connection page. You add custom headers by uploading a schema on the Custom SOAP Headers tab on the Request Headers page and/or Response Headers page, respectively, of the Adapter Endpoint Configuration Wizard.

Add Custom SOAP Headers

Configure custom SOAP headers and specify a valid schema from which an element can be selected as a custom SOAP header in the Adapter Endpoint Configuration Wizard:

1. On the Header page, select Yes for Do you want to configure headers for this Endpoint?.

2. For What types of Headers do you want to configure, select Custom SOAP Headers in the Request Headers and Response Headers columns, as applicable.

3. On the Request page and/or Response page, upload a valid schema from which an element can be selected as a custom SOAP header. Note the following restrictions on schemas to upload.
   - The schema to upload must have a targetNamespace.
   - Endpoints that expect custom SOAP headers without a namespace are not supported.
   - Ensure the schema is defined with elementFormDefault as qualified (for child elements with a namespace prefix) or unqualified (for child elements without a namespace prefix), as required.
   - A schema with dependencies (imports/includes/cross references) is not supported. See Example Schemas.
   - Only one element can be selected as a header from the uploaded schema. To select multiple elements:
– For the same targetNamespace, a schema must be uploaded again for each element.
– For a different targetNamespace, a different schema with required targetNamespace can be used.

4. Select the required element from the list of elements displayed from the uploaded schema.
5. Click Add Header to add the element to the configured headers table.

Delete Custom SOAP Headers

1. On the Request and/or Response page, select the check box next to the required header element in the configured headers table.
2. Click Delete Header to remove the element from the configured headers table.

Example Schemas

The following schema includes a targetNamespace. A prefix for a child is required.

```
<xsd:schema
  elementFormDefault="qualified"
  targetNamespace="http://xmlns.oracle.com/sample"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="headerRoot">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="headerChild" type="xsd:string"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

The following schema includes a targetNamespace. A prefix for a child is not required.

```
<hdr:headerRoot xmlns:hdr=http://xmlns.oracle.com/sample>
  <hdr:headerChild>headerValue</hdr:headerChild>
</hdr:headerRoot>
```
Call an Oracle Fusion Applications Business Intelligence Publisher Report Synchronously

When the SOAP Adapter calls an Oracle Fusion Applications business intelligence publisher (BIP) report synchronously, the report data is always returned as a base64-encoded string in the response. This response can be parsed in the integration with the following approaches:

- Recommended: Use the decodeBase64ToReference mapper function to convert the base64 string to a file reference and use the file reference in a stage file action/FTP Adapter read file operation scenario for further processing.
- Alternative: Write the base64 string using an opaque schema to a file using a stage file action/FTP Adapter write file operation scenario, then read the file using a stage file action/FTP Adapter scenario for further processing.

**Note:**

Upon calling a BIP report synchronously, the request sent to the BIP web service initiates report generation and sends a response back after the report is generated. This can cause latency issues in an Oracle Integration environment.
Troubleshoot the SOAP Adapter

Review the following topics to learn about troubleshooting issues with the SOAP Adapter.

Topics:

• Regenerate the SOAP Adapter Connection After WSDL Definition Updates
• Use the Correct SoapUI Version to Load and Test SOAP Endpoints
• Edit Adapter Connections in Active Integrations
• Multiple SOAP Adapter Invokes with Custom Headers Created Prior to 17.4.3
• SOAP Endpoint Invocation Fails with OSB-380001: mustUnderstand Error
• Specify Connection Property Values with the REST API
• Callback Integrations Fail with a Configured SOAP Action Mismatch Error
• Integrations Fail with SAML Security Policy Selected in Inbound Direction
• Schemas Not Successfully Loaded in Mapper When Using Headers Configured with WSDLs Ending in asmx
• Resolve SOAP Action Mismatch Errors in the SoapUI
• Resolve Exceptions While Invoking Oracle Integration From External Clients
• Connection Error When Using the Incorrect TLS Version
• Extra Information is Included in the Response Headers Returned as Part of the Response Message
• Basic Authentication Fields are Unavailable when Updating a SOAP Adapter Connection after Importing a 16.2.5 Integration
• Unexpected Use of the Suppression Insertion of Timestamp into WS-Security Header Feature in the SOAP Adapter Causes an Unrelated Error Response
• Pre-17.2.5 Integration Failures with an Invoke SOAP Adapter Configured with the Basic Authentication Security Policy and Suppress Timestamp Set to No

Additional integration troubleshooting information is provided. See Troubleshoot Oracle Integration in Using Integrations in Oracle Integration.

Regenerate the SOAP Adapter Connection After WSDL Definition Updates

When a WSDL definition is updated in the connection, perform the following steps described in sequence for the SOAP Adapter to reflect the changes.

Common use cases for regenerating the SOAP Adapter after updating the connection are as follows:

• The updated WSDL definition in the connection is not reflected in the integration.
• Regeneration does not occur for the trigger/invoke connection.

• The Regenerate Artifacts option is not available for an orchestrated integration that contains the trigger/invoke connection.

1. Update and save the SOAP Adapter connection.

2. From the menu at the far right for the updated SOAP Adapter connection, click Refresh Metadata.

3. Click the Information icon for the connection and look for the Last Refresh Status to indicate Complete.

4. Edit the integration and edit the SOAP Adapter trigger or invoke connection to start the Adapter Endpoint Configuration Wizard.

5. Re-navigate through the pages of the wizard until you reach the Summary page, then click Done.

6. Accept the major changes or minor changes warning message (as applicable based on the changes to the connection).

   The changes are reflected in the mapper and Expression Builder.

Use the Correct SoapUI Version to Load and Test SOAP Endpoints

When using the SoapUI to load and test Oracle Integration SOAP endpoints, note that older versions of the SoapUI do not use TLS 1.2 as the default communication protocol, while Oracle Integration SOAP endpoints support only TLS 1.2 for the inbound (trigger) direction. This mismatch results in the following error:

   unable to load wsdl

To avoid this issue, perform one of the following tasks:

• Use a SoapUI version greater than 5.4.0.

• Set the following flag in SoapUI.vmoptions:

   -Dsoapui.https.protocols=TLSv1.2

Edit Adapter Connections in Active Integrations

You can edit adapter connections that are currently in use in active integrations. For the changes to take effect, you must then reactive the impacted integrations. This is the expected behavior.

For example, assume you import and activate an orchestrated integration that includes a SOAP Adapter connection as the initial trigger. You then edit the in-use SOAP Adapter connection on the Connections page and modify the WSDL URL, test the connection, and save it. A message is displayed that warns you to reactivate any integrations using this connection for these changes to take effect.
Multiple SOAP Adapter Invokes with Custom Headers Created Prior to 17.4.3

If you have integrations with multiple SOAP Adapter invokes that have custom headers created prior to 17.4.3, note the following details.

Assume you have the following use case:

• Three different integrations (Flow1, Flow2, and Flow3) use the SOAP Adapter as a trigger connection with custom HTTP headers configured (created before 17.4.3). A static namespace is used for the inbound wrapper.

• Flow4 is a delegating pattern integration with multiple invoke connections using the endpoints of Flow1, Flow2 and Flow3 (created in 18.3.3). This flow uses dynamic namespaces for all outbound wrappers.

Activating Flow4 causes a failure because there is a conflict with the wrapper schema between Flow1 and Flow2 when used as an invoke.

Note the following workaround and solutions:

• Workaround:
  Ensure that the schema is the same between integrations (for example, Flow1 and Flow2). If the custom HTTP headers used have a different description, this can cause a schema conflict. Keep the same description in all integrations for the schema to be synchronized. If you edit the Flow2 trigger description to match Flow1 and re-activate the integration, then recreate the Flow2 invoke in Flow4, the issue is resolved.

• Actual solution:
  Recreate the Flow1, Flow2, and Flow3 integrations to use the dynamic wrapper namespaces that are available after 17.4.3. If the flows are recreated in the latest Oracle Integration version, dynamic namespaces are available and there is no activation issue for Flow4. Note that there is significant rework required to recreate the integrations because it involves some complex logic and the change is in the trigger.

• Alternative solution to avoid rework:
  Manually update the namespaces in the Flow1, Flow2, and Flow3 triggers to include dynamic namespaces and reactivate the integrations, then use them in Flow4.

Note:
Manual updating is not recommended because the change can impact downstream mappings and assignments based on the integration design.
SOAP Endpoint Invocation Fails with OSB-380001: mustUnderstand Error

If SOAP endpoint invocation fails in an integration with the following OSB-380001 - mustUnderstand fault, there are several possible solutions.

```xml
<fault xmlns="http://www.bea.com/wli/sb/context">
<errorCode>OSB-380001</errorCode>
<reason>Fault received on invocation of target : https://host:port/endpoint
<![CDATA[ Fault Code : codeNS:MustUnderstand Fault String : Unprocessed 'mustUnderstand' header element:
(http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd)
Security ||> </reason>
<location>node>RouteNode1</node><path>response-pipeline</path>
</location>
</fault>
```

Solutions:

- If the SOAP Adapter connection was created with a static WSDL file or dynamic WSDL URL (valid URL), but the connection fails at runtime with the following error/fault message pointing to an unexpected host/port or endpoint URL, ensure that the WSDL has the correct service endpoint URL.

```xml
<service name="PrecisionService">
  <port binding="tns:PrecisionServiceBinding" name="PrecisionPort">
    <soap:address location="http://correct_host:correct_port/correct_soap_endpoint"/>
  </port>
</service>
```

- If the WSDL has incorrect endpoint information as seen in the fault message, fix the WSDL service port to resolve the issue.

Specify Connection Property Values with the REST API

If using the REST API to specify SOAP Adapter connection properties, the property values to specify are different than those specified in the Connections page of Oracle Integration. If you incorrectly specify these values, the security policy is changed and you receive runtime errors.

To correctly configure these properties, specify the following values as JSON input:
Callback Integrations Fail with a Configured SOAP Action Mismatch Error

A callback integration fails with a configured SOAP action mismatch error when the trigger in the callback integration is configured with a connection using the Upload File checkbox to upload a WSDL that does not have a binding section.

As a workaround, change Disable SOAP Action Validation to Yes on the Operations page for the trigger configuration and reactivate the flow.
Integrations Fail with SAML Security Policy Selected in Inbound Direction

Verify if the target service issuer certificates (also called DemoCA or CloudCA) are imported into Oracle Integration using the **Message Protection Certificate** option.
Verify if the SAML user is configured in the Oracle Integration My Services Dashboard page.

Schemas Not Successfully Loaded in Mapper When Using Headers Configured with WSDLs Ending in asmx

If headers are configured in the SOAP Adapter to use Dot Net or Microsoft WCF-based services (a WSDL that generally ends with asmx), the schemas are not successfully loaded in the mapper because the prefixes for schema and prefix s4 for the element type are not defined at the schema level.

The schemas are in line to the WSDL with prefix declarations at the WSDL level, but not at the schema level. The following code sample is from the WSDL:

```xml
  targetNamespace="http://tempuri.org/
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  ....
  <s:schema elementFormDefault="qualified" targetNamespace="http://webservices.com/1.0/Core/"
    xmlns:s="http://www.w3.org/2001/XMLSchema"
    xmlns:s4="http://webservices.com/1.0/Core/">
    ....
  </s:schema>
</wsdl:Definitions>
```

As a workaround add the prefix declarations manually in the schemas, re-import the IAR file and proceed. The following is an example of the schema in the WSDL after artifact generation:

```xml
<s:schema elementFormDefault="qualified"
  targetNamespace="http://webservices.com/1.0/Core/"
  xmlns:s="http://www.w3.org/2001/XMLSchema"
  xmlns:s4="http://webservices.com/1.0/Core/">
  ....
</s:schema>
```

Resolve SOAP Action Mismatch Errors in the SoapUI

If you have a SOAP action mismatch, a 500 Internal server error occurs while invoking the integration from the SoapUI. For example, you may have a Salesforce Adapter configured with an empty SOAP action and expecting an empty SOAP action from the client. However, clients such as the SoapUI generate requests with a SOAP
action identifier read/available from the WSDL file. To resolve this error, update the default SOAP action with SOAP headers that include empty values.

1. Start the SoapUI.
2. Import the WSDL.
3. Click the Header tab at the bottom of the page.
4. Click the + button to create a header.
   The Add HTTP Header dialog is displayed.
5. Specify the name of the header to add (for example, SoapAction).
6. Update the value for the SOAP action header. For this example, the Value field is left empty to match the Salesforce Adapter.
7. Verify the updated headers by clicking the Raw tab.

Resolve Exceptions While Invoking Oracle Integration From External Clients

If you receive an exception error while invoking Oracle Integration from an external client (for example, SOAP UI), Oracle Integration requires a WSS username token. The WSS username token must be passed from an external client such as the SOAP UI.

A sample header with a WSS username token looks as follows:

```xml
<wsse:Security soapenv:mustUnderstand="1" xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
 xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
  <wsu:Timestamp wsu:Id="TS-135B2A61F1A7AB1C39149198776769232">
    <wsu:Created>2017-04-12T09:01:16.922Z</wsu:Created>
    <wsu:Expires>2017-04-12T09:02:16.922Z</wsu:Expires>
  </wsu:Timestamp>
  <wsse:UsernameToken wsu:Id="UsernameToken-135B2A61F1A7AB1C391491987767692101">
    <wsse:Username>weblogic</wsse:Username>
  </wsse:UsernameToken>
</wsse:Security>
```
To set the WSS username token:

1. Start the SOAP UI.
2. Click the AUTH tab.
3. Select Basic from the Add Authorization dialog, and click OK.
4. Complete the authorization credentials.
5. Right-click the payload and select the Add WSS Username Token and Add WSS–Timestamp options.
6. Specify the time in the Specify Time-to-Live value dialog, and click OK.

Connection Error When Using the Incorrect TLS Version

If you receive the following design time or runtime error and you have already imported your SSL certificate into Oracle Integration, ensure that you are using the correct TLS version.

Design Time Error:

javax.net.ssl.SSLHandshakeException: Remote host closed connection during handshake
or
java.net.SocketException: Connection reset
or
Caused by: javax.net.ssl.SSLHandshakeException: Remote host closed connection during handshake
at sun.security.ssl.SSLSocketImpl.readRecord(SSLSocketImpl.java:946)
at
sun.security.ssl.SSLSocketImpl.performInitialHandshake(SSLSocketImpl.java:1312)
} at
sun.security.ssl.SSLSocketImpl.startHandshake(SSLSocketImpl.java:1323)
at
....
.....
Extra Information is Included in the Response Headers Returned as Part of the Response Message

When standard HTTP headers are used in the response headers as part of the response message, extra information is included as part of the returned output data at runtime. This issue occurs only when headers are used in the SOAP Adapter. Without headers, the output is returned without extra information in the response message.

The extra information is not an extra namespace. It is a valid namespace matching the prefix of the element. Without headers, the namespace comes as an attribute in the root element. With headers, since elements are converted from a wrapper, the child elements are copied along with their namespaces.

For example, with a connection that uses the same web service, but one with a header (custom HTTP Header) and another one without a header, the body elements in the response message are different as shown below:

**Without a header:**

```xml
<nstrgmprr:result>
    <nsmpr6:PartyId>10</nsmpr6:PartyId>
    <nsmpr6:PartyName>Acme Corp</nsmpr6:PartyName>
</nstrgmprr:result>
```

**With a header:**

```xml
<nsmprr2:result>
    <nsmprr8:PartyId xmlns:nsmprr8="http://xmlns.oracle.com/apps/cdm/foundation/parties/organizationService ">
        10
    </nsmprr8:PartyId>
    
    <nsmprr8:PartyName xmlns:nsmprr8="http://xmlns.oracle.com/apps/cdm/foundation/parties/organizationService ">
        Acme Corp
    </nsmprr8:PartyName>
</nsmprr2:result>
```
Basic Authentication Fields are Unavailable when Updating a SOAP Adapter Connection after Importing a 16.2.5 Integration

After importing an IAR file that was exported from a 16.2.5 release, while updating a SOAP Adapter Connection that uses Basic Authentication, the Username and Password fields are not available in the Configure Security dialog. For a trigger connection, these credentials are not required because this is a SOAP endpoint. The credentials to access the endpoint are always the Oracle Integration runtime user and password. In previous releases, the inbound SOAP Adapter connection accepted username and password changes.

Unexpected Use of the Suppression Insertion of Timestamp into WS-Security Header Feature in the SOAP Adapter Causes an Unrelated Error Response

When creating a SOAP Adapter connection with Suppress insertion of timestamp into WS-Security header set to Yes on the Connections page and using a WSDL that requires a timestamp header, the connection fails during runtime with the following error instead of a message indicating that a valid timestamp is not present:

<errorCode>OSB-380001</errorCode>
<reason>InvalidSecurityToken : The security token is not valid.</reason>

Pre-17.2.5 Integration Failures with an Invoke SOAP Adapter Configured with the Basic Authentication Security Policy and Suppress Timestamp Set to No

Integrations created prior to 17.2.5 that include an invoke SOAP Adapter configured with the Basic Authentication security policy and Suppress Timestamp set to No fail at runtime in 17.2.5 with a security error mentioning mustUnderstand. This indicates that the service is not expecting a timestamp in the request.

As a workaround, deactivate the flow, update the connection with Suppress Timestamp set to Yes, save the connection, and activate the flow.