DecisionPoint 3

Business Service Connectivity Guide

veda
applied intelligence

Contents

1 Introduction

The purpose of this document is to provide an overview of the general rules and guidelines to be followed when connecting to a DecisionPoint3 / DataLink Manager (DLM) Business Service. A DP3/DLM Business Service is deployed to a DecisionPoint3/DLM Business Service Manager (BSM).

The document will cover the following:

- Basic overview of each service operation
- General connectivity guidelines
- Sending requests to services
- Reading responses
- Troubleshooting and understanding error messages

The target audience are developers/support staff who, given a Business Service endpoint, need to connect to it and execute one or more operations.

For most scenarios throughout this document there will be given specific examples. For this purpose we will be using a single deployed Business Service, appropriately named – SampleService.

Note:
The business rules and use case details behind each service operation are not in the scope of this document. The sample XML is provided to give readers insight to the services and method available in DecisionPoint3/DLM and will vary for each implementation.

2 Operations Overview

The following section will provide a brief overview of all operations available on a deployed Business Service. The main purpose is to provide some basic familiarity with the various operations and a meaningful context to the example requests/responses given throughout the document.

Operation	*Description*
Ping	Ping is a utility operation with its main purpose being providing the opportunity to test and verify that the service is up and running.
GenerateId	The GenerateId operation is used to generate a unique ID which can then later be used in the Submit/SubmitWait operations.
Submit	The Submit operation allows the client to **asynchronously** submit an transaction. The service responds immediately, providing a signal that the transaction has been queued for processing. To get the result of the processing, the Retrieve operation should be invoked.
SubmitWait	The SubmitWait operation allows the client to **synchronously** submit a transaction, i.e. the client waits until the service responds with the result of processing the transaction.
Retrieve	The Retrieve operation returns a transaction for a given transaction id.
Save	The Save operation allows the client to save a transaction to the BSM database for future processing.
SearchMetaData	The SearchMetaData operation allows the user to search for a transaction using the core transaction or metadata fields defined by the user in the Studio designer.
GetCacheEntryList	The GetCacheEntryList operation provides a list of connector cache entry id's for the given transaction id. These are the connector cache entries that were used while processing the particular transaction.
GetCacheEntry	This operation is used for retrieving the actual connector cache entry content based on the cache id's returned from the GetCacheEntryList service.
User-Defined Functions (UDFs)	Apart from the already mentioned operations which are standard for every Business Service, there can also be a variable number of other user-defined operations. These operations are defined by the path designer and they are not in the scope of this document.

3 Connectivity Overview

The business functionality of a deployed DecisionPoint3/DLM Business Service is provided via a **Web Service**[1]. Each service operation therefore is a **WebMethod**.

The following sections will give an overview of the various protocols used in implementing these web services, as well as security-related information.

3.1 Binding Protocol

The binding protocol used for BSM web services is **Simple Object Access Protocol** (**SOAP**), version **1.1**. The SOAP messages are routed over **HTTP (Hypertext Transfer Protocol) or HTTPS (HTTP Secure)** which ensures interoperability as the client and the service providers can be running on different platforms.

The version means that any client requests must be compatible with the SOAP 1.1 format, otherwise the server will reject the request with an appropriate error (See

SOAP Mismatch).

3.2 SOAP

SOAP is a protocol for exchanging XML-based messages over a network, typically using HTTP protocol. The SOAP message format is comprised of a SOAP Envelope which encloses all request information. The SOAP Envelope, in turn, is then made up of optional headers and a body. The headers optionally contain context related information, such as security, while the body contains actual payload or transaction data.

All messages sent to the BSM web services must adhere to the SOAP 1.1 standard. The BSM supports the **Document** style of SOAP messages (deals with XML documents as payloads which adhere to XML schema definitions) with **literal** binding (meaning plain text messages without encoding) in order to achieve high interoperability. Many examples of SOAP 1.1 compliant messages are provided throughout this document.

3.3 Transport Protocol

[1] http://www.w3.org/TR/ws-arch/

As mentioned above, the transport protocol used in routing the SOAP messages between client and server is HTTP/HTTPS. BSM supports both versions and switching from one to another can easily be done by the BSM administrator.

HTTPS creates a secure transport layer over the standard insecure HTTP by combining it with the **SSL (Secure Socket Layer) 3.0** protocol and as a result provides encrypted communication and reasonable protection from eavesdropping and "man-in-the-middle" attacks.

3.4 Security

3.4.1 Authentication

Apart from the above explained HTTPS transport protocol, the BSM web services are additionally secured via **Web Services Security**[2] .

All web services are secured via UsernameToken Profile 1.1[3] including Nonce and Security Timestamp validation[4]. If the username/password combination is invalid or the time of request is after the expiry date the service will return an error (See Failed Authentication)

3.4.2 Authorisation

Apart from authenticating the user, the requests are also validated by means of authorisation, i.e. determining if the user has the correct role that allows him to execute the particular service.

The roles and access rights are user-configurable (Studio is used for defining the roles that can execute a service and the BSM Admin Console for mapping users to roles). In case a user does not have the correct role, the service will return an error (See Failed Authorisation).

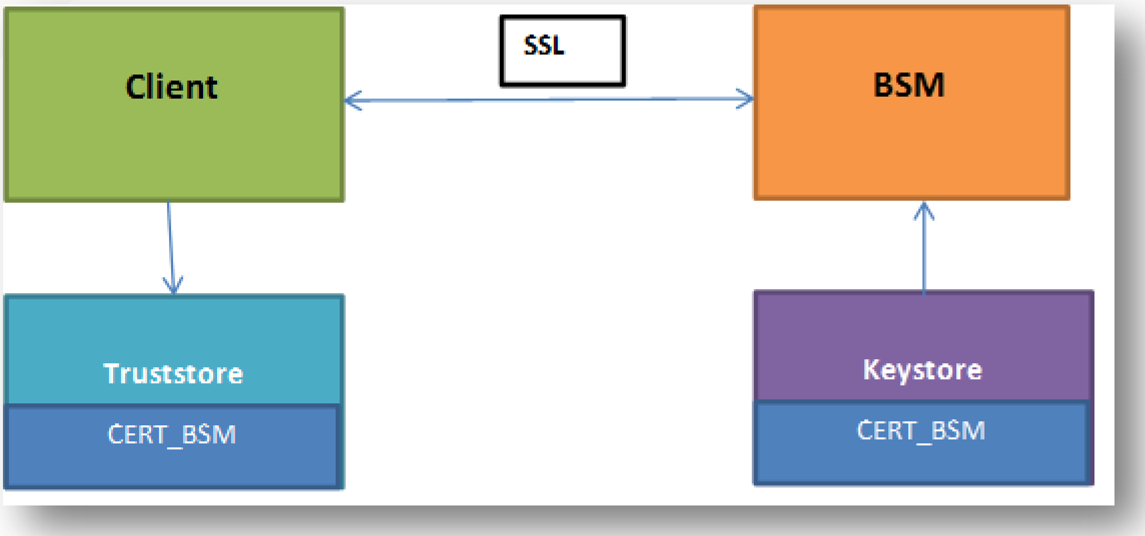
3.4.3 SSL and Certificates

If using HTTPS as the transport protocol, the BSM and deployed services support all three standard SSL and certificate verification configurations. A brief description of all three is provided in the following sections.

3.4.3.1 Self-signed SSL

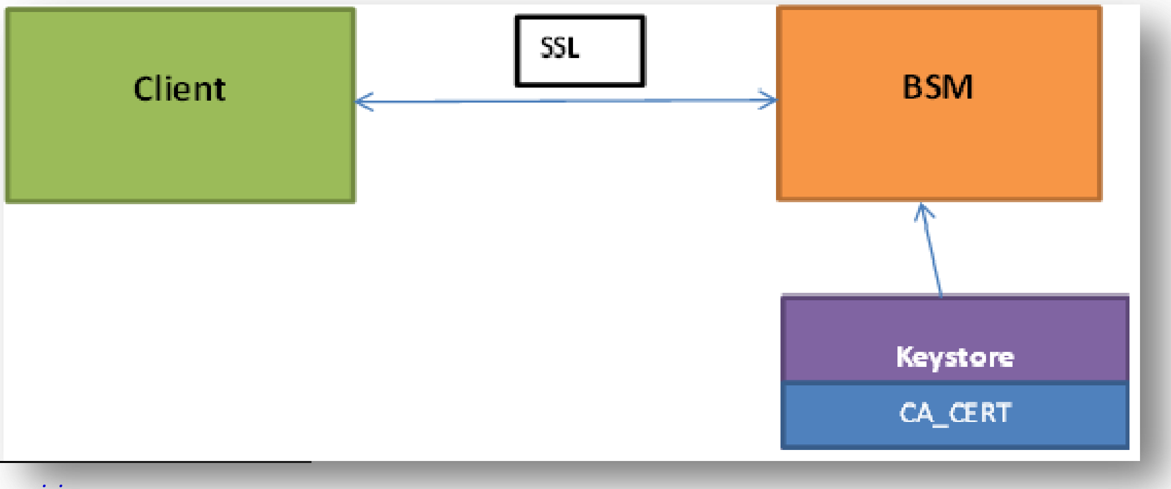
This is the default set-up for BSM. The server has a self-signed certificate which the client needs to trust in order to establish a connection. In other words, it must keep the server's certificate in its trust store.

[2] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss
[3] http://www.oasis-open.org/committees/download.php/16782/wss-v1.1-spec-os-UsernameTokenProfile.pdf
[4] https://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf



3.4.3.2 Commercially signed SSL Certificate

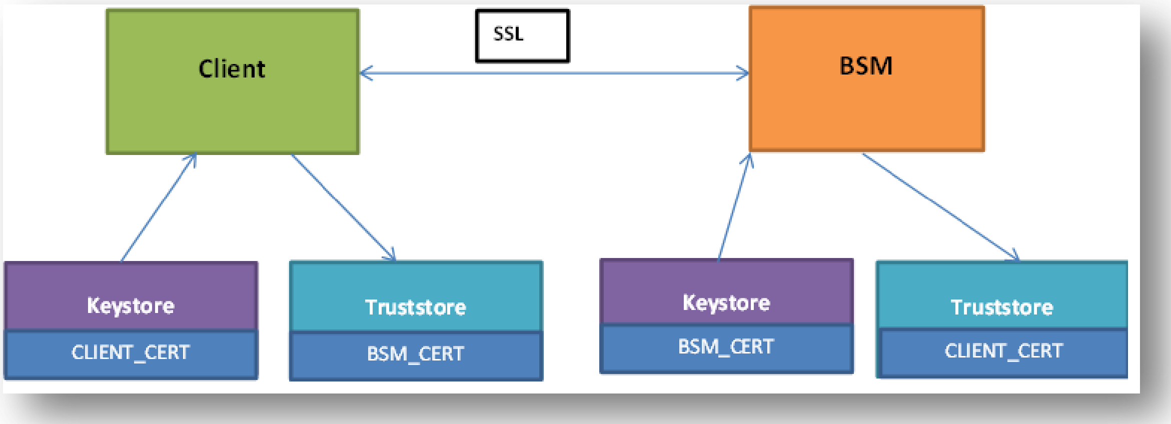
This option is available when there is a need to have a commercially signed certificate which will be automatically trusted by the client. This also removes the need for adding the certificate to the client's trust store. Examples of providers of such commercial certificates are VeriSign[5] , DigiCert[6] etc.



[5] www.verisign.com.au
[6] www.digicert.com

3.4.3.3 SSL with Mutual Authentication

This SSL configuration requires both the server and client to prove their identity to each other and each maintain their own key and trust stores. BSM will support this by simply adding a trust store file during start up.



4 Connecting

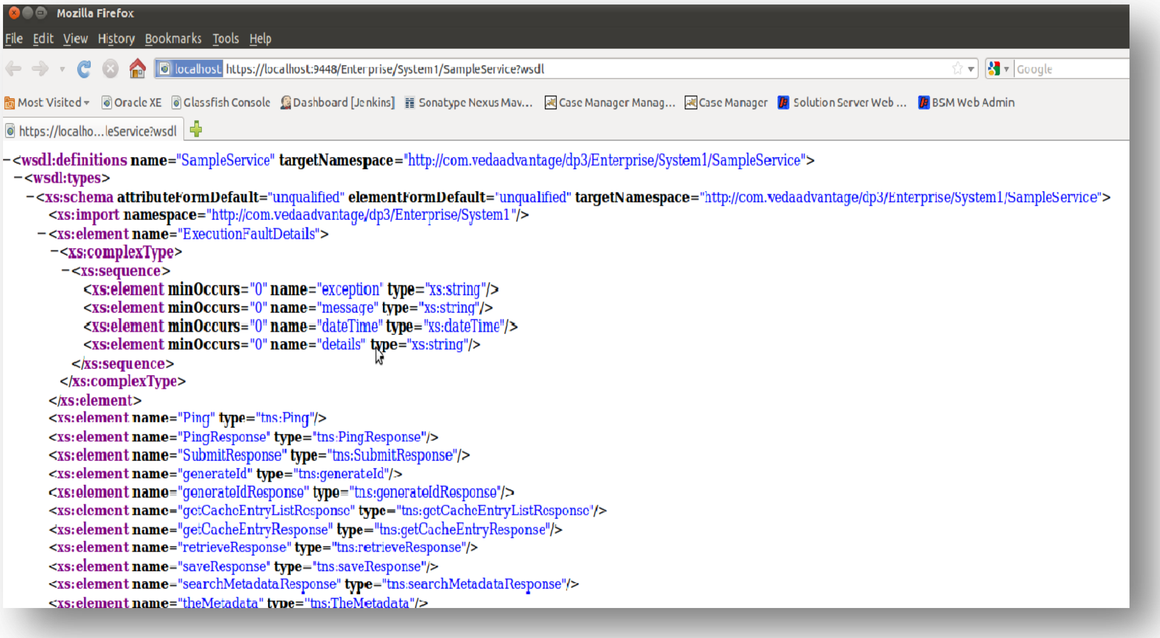
The following section will take you through the basic guidelines for building valid Business Service requests.

4.1 Retrieving the Service WSDL

Whether you want to test the service connectivity with one of the available free tools (such as SoapUI – see Test Tools), build your client to connect to the desired Business Service or simply view the service WebMethods and their associated data types, you will need to retrieve the service WSDL (Web Service Definition Language) file.

Assuming you have the service endpoint available, that is as easy as adding a "**?wsdl**" suffix to the given endpoint. If you do this in your browser, it will show you the full WSDL with all operations (WebMethods) and data types described.

The following figure shows part of the WSDL for our SampleService.



```
-<wsdl:definitions name="SampleService" targetNamespace="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService">
 -<wsdl:types>
  -<xs:schema attributeFormDefault="unqualified" elementFormDefault="unqualified" targetNamespace="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService">
     <xs:import namespace="http://com.vedaadvantage/dp3/Enterprise/System1"/>
   -<xs:element name="ExecutionFaultDetails">
     -<xs:complexType>
       -<xs:sequence>
          <xs:element minOccurs="0" name="exception" type="xs:string"/>
          <xs:element minOccurs="0" name="message" type="xs:string"/>
          <xs:element minOccurs="0" name="dateTime" type="xs:dateTime"/>
          <xs:element minOccurs="0" name="details" type="xs:string"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="Ping" type="tns:Ping"/>
    <xs:element name="PingResponse" type="tns:PingResponse"/>
    <xs:element name="SubmitResponse" type="tns:SubmitResponse"/>
    <xs:element name="generateId" type="tns:generateId"/>
    <xs:element name="generateIdResponse" type="tns:generateIdResponse"/>
    <xs:element name="getCacheEntryListResponse" type="tns:getCacheEntryListResponse"/>
    <xs:element name="getCacheEntryResponse" type="tns:getCacheEntryResponse"/>
    <xs:element name="retrieveResponse" type="tns:retrieveResponse"/>
    <xs:element name="saveResponse" type="tns:saveResponse"/>
    <xs:element name="searchMetadataResponse" type="tns:searchMetadataResponse"/>
    <xs:element name="theMetadata" type="tns:TheMetadata"/>
```

The WSDL provides all the basic information needed to successfully connect to the service as well as to build requests for the various WebMethods.

4.2 Test Tools

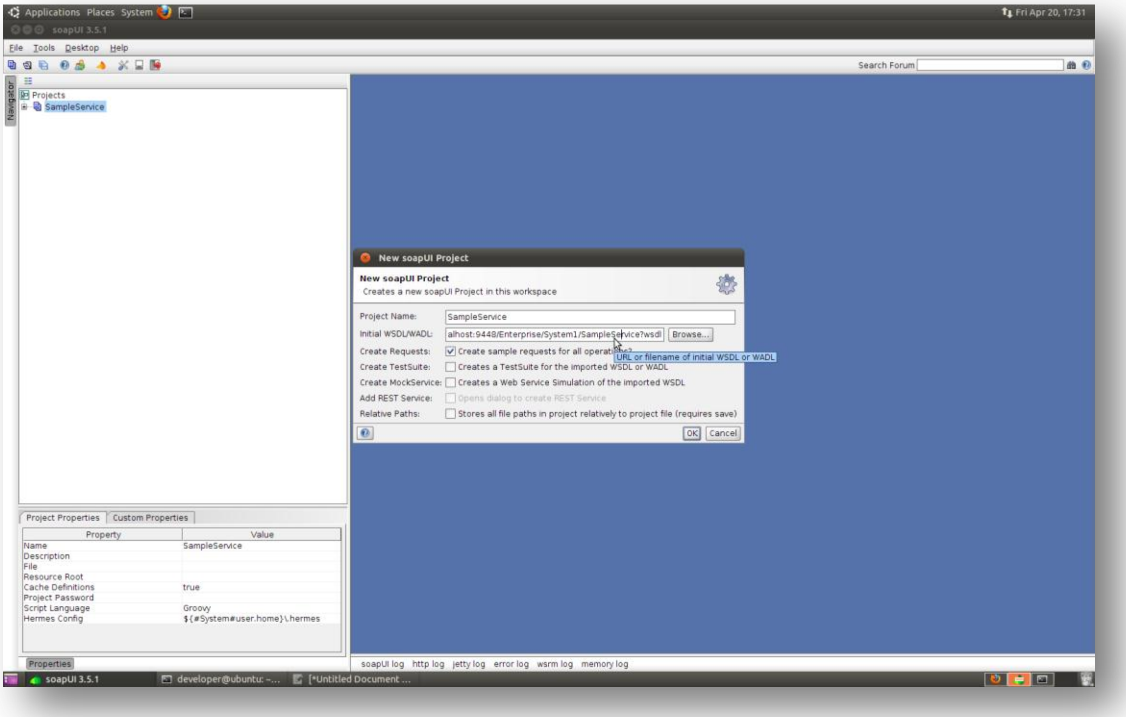
Before you build your client for connecting to a DP3/DLM Business Service, you can increase your familiarity and test your requests by using some of the available free tools.

4.2.1 SoapUI

One of those tools is SoapUI[7] and is the tool used for creating the requests and documenting the responses throughout this document.

4.2.1.1 Creating a New Project

Creating a new project in SoapUI is simple – go to *File->New SoapUI Project* and a dialog will appear asking you for the project name and WSDL. You can provide whichever name you want while the WSDL will, as mentioned earlier, be the endpoint of your Business Service with the "**?wsdl**" suffix.



New soapUI Project

New soapUI Project
Creates a new soapUI Project in this workspace

Project Name: SampleService
Initial WSDL/WADL: alhost:9448/Enterprise/System1/SampleService?wsdl Browse...
Create Requests: ☑ Create sample requests for all operati...
URL or filename of initial WSDL or WADL
Create TestSuite: □ Creates a TestSuite for the imported WSDL or WADL
Create MockService: □ Creates a Web Service Simulation of the imported WSDL
Add REST Service: □ Opens dialog to create REST Service
Relative Paths: □ Stores all file paths in project relatively to project file (requires save)

OK Cancel

Project Properties | Custom Properties

Property	Value
Name	SampleService
Description	
File	
Resource Root	
Cache Definitions	true
Project Password	
Script Language	Groovy
Hermes Config	${#System#user.home}\.hermes

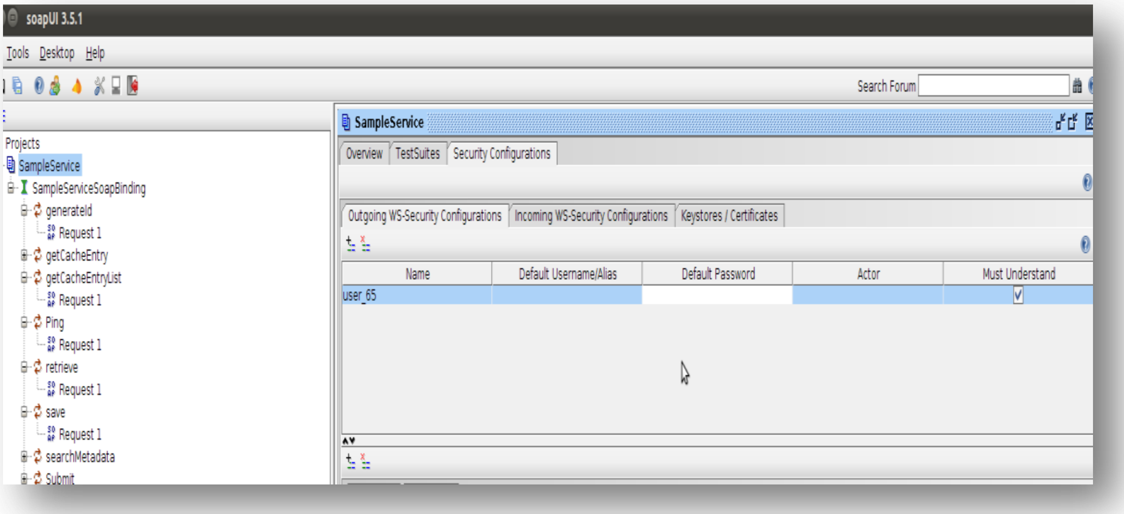
This will create a new project with all the Business Service WebMethods available.

[7] http://www.soapui.org/

4.2.1.2 Security Configuration

As mentioned earlier, all Business Services are secured and require the correct user credentials to be executed as well as the timestamp information (TTL). In SoapUI, you can configure these parameters via the Security Configurations tab for each service.

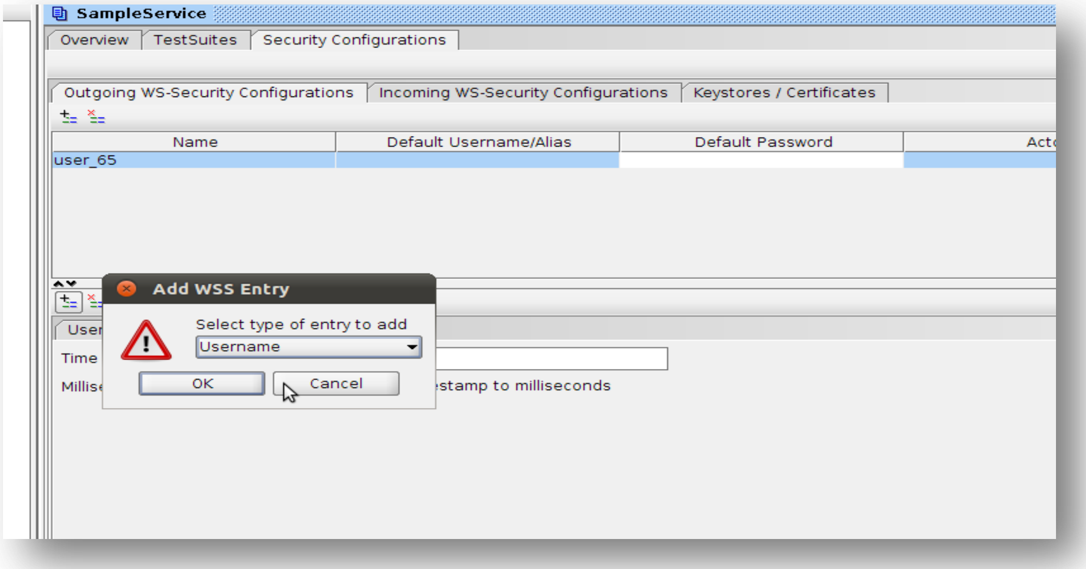
To configure security details, double-click on the project name. A dialog will appear, select the *Security Configurations* tab and then *Outgoing WS-Security Configurations*.



Name	Default Username/Alias	Default Password	Actor	Must Understand
user_65				☑

Add a user by clicking the ‘+’ button on the top panel and enter an identifier for the user (user_65 in the above figure). Now you need to add a Web Service Security (WSS) entry so click the ‘+’ on the bottom panel.

A dialog will appear, select *Username* and click OK.

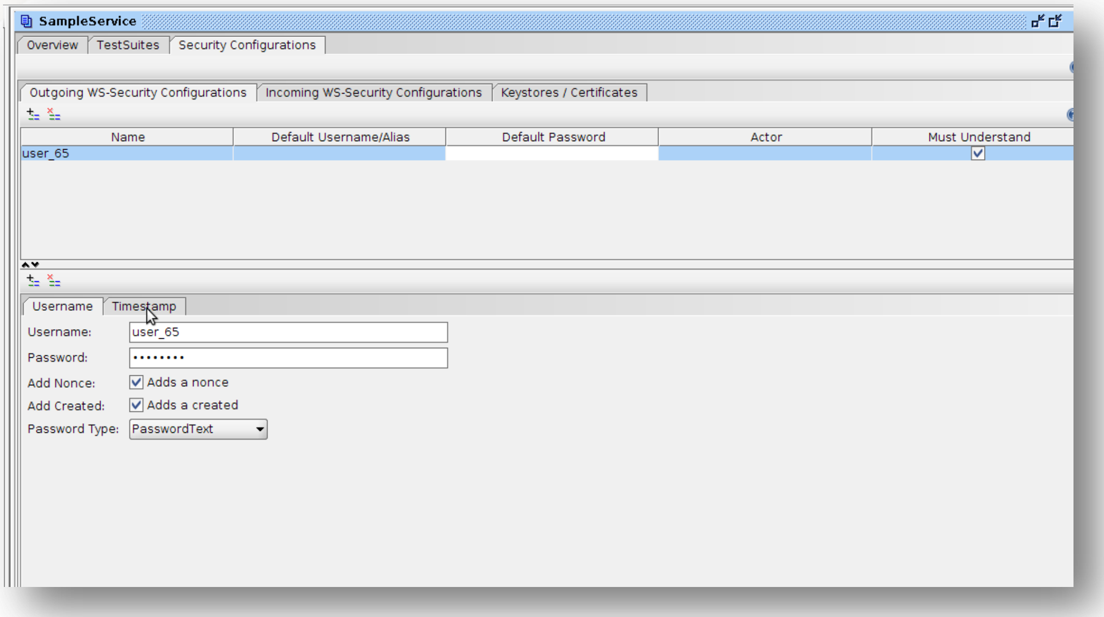


Name	Default Username/Alias	Default Password	Act
user_65			

Add WSS Entry

Select type of entry to add: Username

After you’ve done this, you need to enter the user name and password in the bottom panel. Don’t forget to select **PasswordText** as the Password Type.



SampleService

Overview | TestSuites | Security Configurations

Outgoing WS-Security Configurations | Incoming WS-Security Configurations | Keystores / Certificates

Name	Default Username/Alias	Default Password	Actor	Must Understand
user_65				☑

Username | Timestamp

Username: user_65

Password: ••••••••

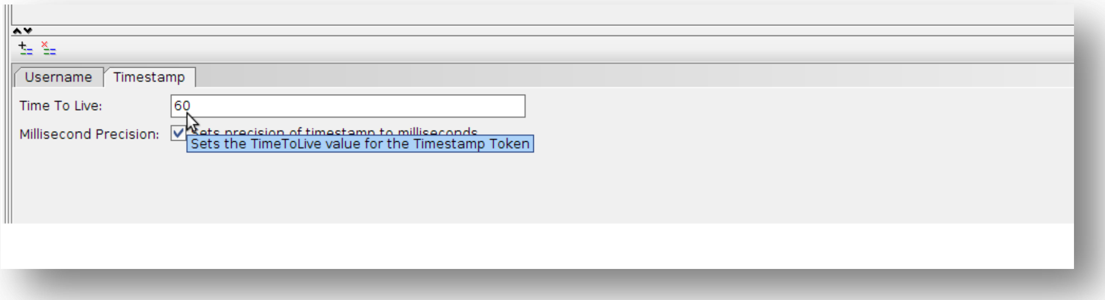
Add Nonce: ☑ Adds a nonce

Add Created: ☑ Adds a created

Password Type: PasswordText

After you’ve done this, you have completed the user credentials part of the security requirements to execute a DP3/DLM Business Service. Now you also need to add the timestamp configuration.

You can do this by adding a new WSS entry so click the ‘+’ button on the bottom panel again, but this time instead of Username choose Timestamp and click OK. Set a time in milliseconds for the request to be valid, normally a figure of around 60 milliseconds should prove sufficient.



Username | Timestamp

Time To Live: 60

Millisecond Precision: ☑ Sets precision of timestamp to milliseconds

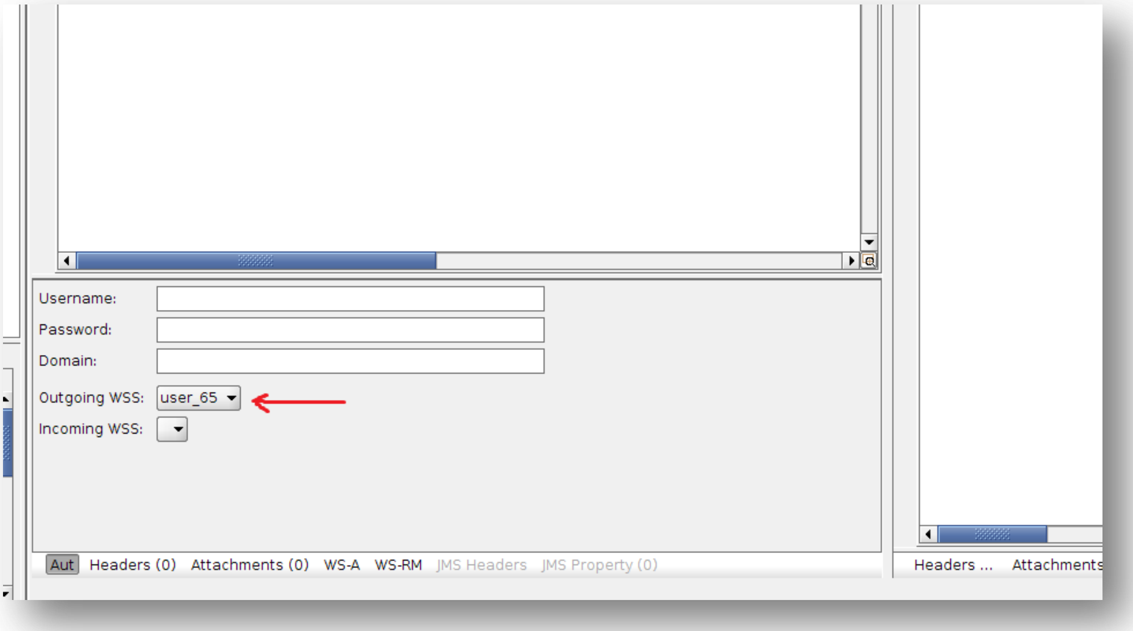
Sets the TimeToLive value for the Timestamp Token

This step finalizes the security configuration for your new SoapUI project.

4.2.1.3 Testing Requests

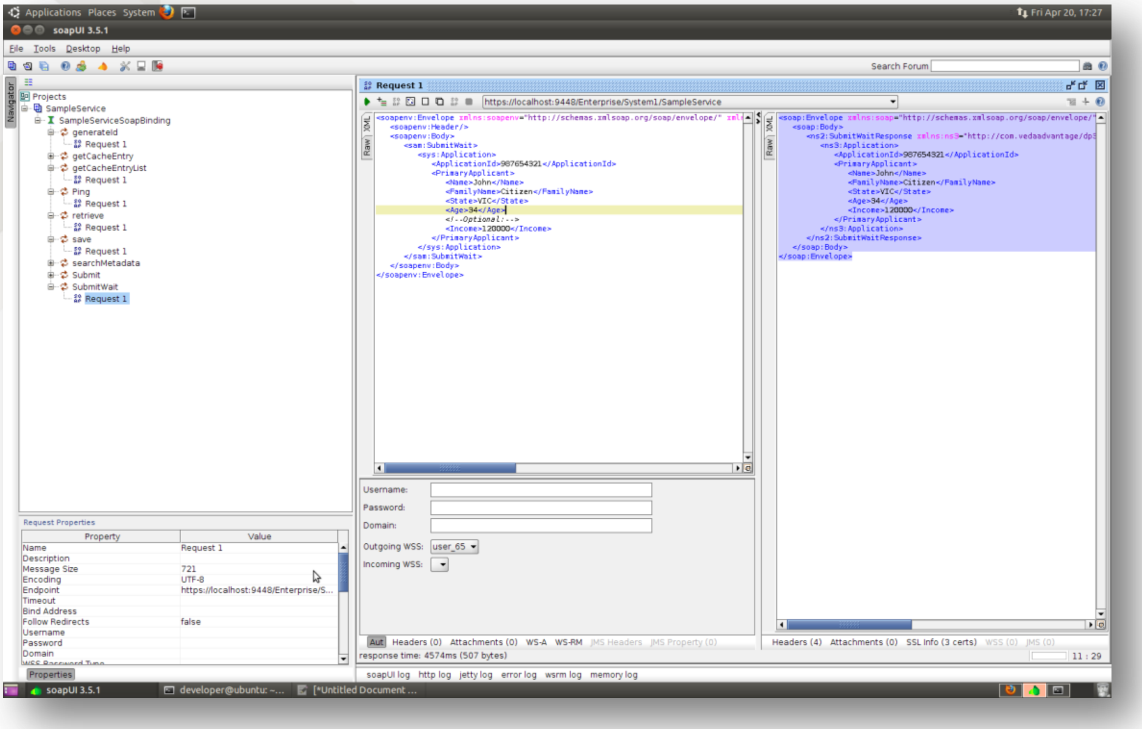
You can now view all the operations, see their input/output types as well as build new requests manually by populating the template requests provided by SoapUI.

Note:
Don't forget to select the user whose credentials you wish to be included in the security header of your request. You can do that at the bottom of the Request panel, by selecting Aut->Outgoing WSS and then selecting one of your pre-configured users.



Username:
Password:
Domain:
Outgoing WSS: user_65
Incoming WSS:

You can now execute the service by clicking the "Play" button in the request builder and the response will appear in the RHS of the window.



https://localhost:9448/Enterprise/System1/SampleService

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmln
   <soapenv:Header/>
   <soapenv:Body>
      <sam:SubmitWait>
         <sys:Application>
            <ApplicationId>987654321</ApplicationId>
            <PrimaryApplicant>
               <Name>John</Name>
               <FamilyName>Citizen</FamilyName>
               <State>VIC</State>
               <Age>34</Age>
               <!--Optional:-->
               <Income>120000</Income>
            </PrimaryApplicant>
         </sys:Application>
      </sam:SubmitWait>
   </soapenv:Body>
</soapenv:Envelope>
```

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
   <soap:Body>
      <ns2:SubmitWaitResponse xmlns:ns3="http://com.vedaadvantage/dp3
         <ns3:Application>
            <ApplicationId>987654321</ApplicationId>
            <PrimaryApplicant>
               <Name>John</Name>
               <FamilyName>Citizen</FamilyName>
               <State>VIC</State>
               <Age>34</Age>
               <Income>120000</Income>
            </PrimaryApplicant>
         </ns3:Application>
      </ns2:SubmitWaitResponse>
   </soap:Body>
</soap:Envelope>
```

Outgoing WSS: user_65

response time: 4574ms (507 bytes)

Request Properties

Property	Value
Name	Request 1
Description	
Message Size	721
Encoding	UTF-8
Endpoint	https://localhost:9448/Enterprise/S...
Timeout	
Bind Address	
Follow Redirects	false
Username	
Password	
Domain	

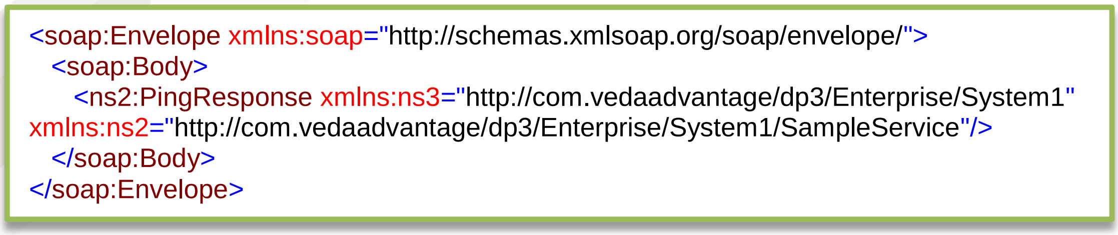
4.3 Testing Service Availability

Before you execute any service requests, you might want to ensure that the service is up and running. For this purpose, every deployed Business Service has a Ping WebMethod available. If the Ping successfully returns with an empty response, the service is running and ready to accept client requests.

The following shows a request to our SampleService's Ping operation:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sam="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService">
   <soapenv:Header/>
   <soapenv:Body>
      <sam:Ping/>
   </soapenv:Body>
</soapenv:Envelope>
```

If the service is up and running we can expect a response like the following:



```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns2:PingResponse xmlns:ns3="http://com.vedaadvantage/dp3/Enterprise/System1"
xmlns:ns2="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService"/>
  </soap:Body>
</soap:Envelope>
```

If the service is not available connection to the server will be refused.

4.4 Building Requests

This chapter describes the guidelines for building a request. Each request is written as a standard XML document.
XML elements are logically grouped under a single root element and each element may contain zero or more other child elements.
For a request to be successfully processed, it must pass the validation process. The request:

1. **Must be in a valid XML format**

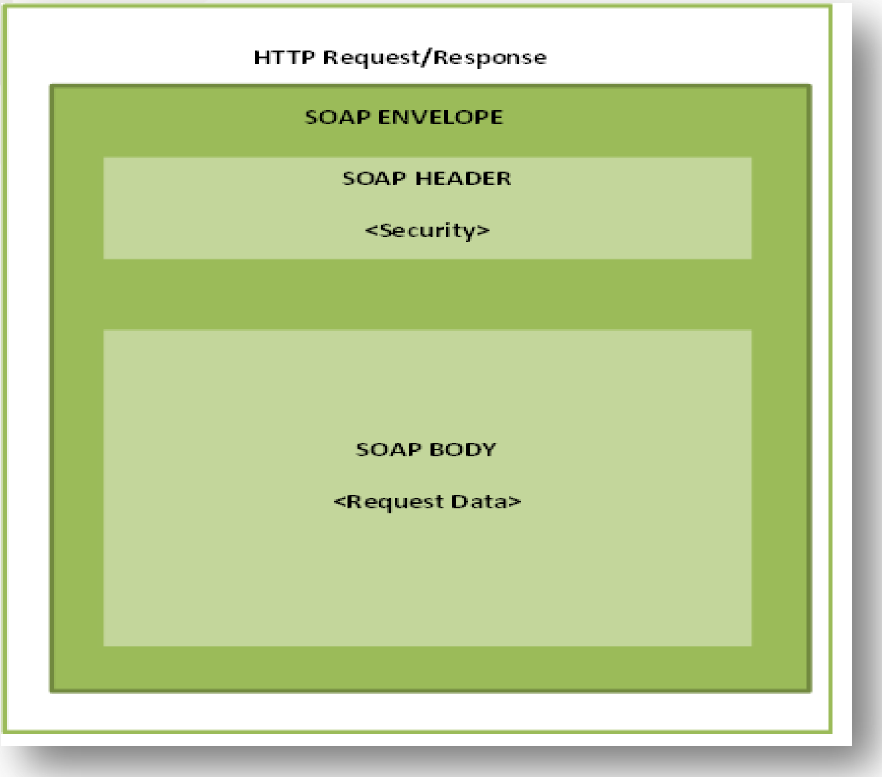
2. **Must be a valid SOAP 1.1 request**

3. **Must have valid user credentials and nonce information included**

4. **Must adhere to the schema as defined in the WSDL for the particular service**

5. **Must adhere to any further field restrictions as specified during the service (path) design process**

As mentioned above, your request message must be formulate as a standard SOAP 1.1 document. The standard layout of a SOAP request is outlined in the following figure.



The key features of a BSM service request message are:

- The Header uses the UsernameToken 1.1 Profile + Security Timestamp.
- The Body uses document style messaging and thus only contains a single request payload (<Request Data>)

The <Request Data> section contains all the information transmitted to the service. It consists of elements containing data specific to the associated service. The data elements provided must match the structure, type and restrictions described in the relevant WSDL document. The response from BSM web services is also in a SOAP 1.1 compliant format.

The following sections provide request/response examples for each Web Method available for a deployed Business Service.

4.4.1 Submit and SubmitWait

The structure for these two operations will vary between Business Services, depending on the input/output types specified by the path designers. Both operations will have the same input type with the only difference being the operation name.

```
<soapenv:Envelope xmlns:sam="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sys="http://com.vedaadvantage/dp3/Enterprise/System1">
  <soapenv:Header>

<wsse:Security soapenv:mustUnderstand="1" xmlns:wsse="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
<wsu:Timestamp wsu:Id="Timestamp-10" xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
   <wsu:Created>2012-04-12T07:16:35.068Z</wsu:Created>
   <wsu:Expires>2012-04-12T07:17:35.068Z</wsu:Expires>
</wsu:Timestamp>
<wsse:UsernameToken wsu:Id="UsernameToken-9" xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
            <wsse:Username>john</wsse:Username>
             <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
                      username-token-profile-1.0#PasswordText">sfsdfsdfsfsfs</wsse:Password>
            <wsse:Nonce EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
soap-message-security-1.0#Base64Binary">w0eqr+OjrV4OcJLYfWPOWQ==</wsse:Nonce>
          <wsu:Created>2012-04-12T07:16:35.068Z </wsu:Created>
</wsse:UsernameToken>
</wsse:Security>
</soapenv:Header>
  <soapenv:Body>
    <sam:Submit>
      <sys:Transaction>
        <TransactionId>12345</TransactionId>
        <PrimaryApplicant>
          <Name>John</Name>
          <FamilyName>Citizen</FamilyName>
          <State>VIC</State>
          <Age>34</Age>
          <Income>120000</Income>
        </PrimaryApplicant>
      </sys:Transaction>
    </sam:Submit>
  </soapenv:Body>
</soapenv:Envelope>
```

Note:

The data between the <sys:Transaction> tags will vary between Business Services, as explained earlier.

A successful Submit operation will return an empty response.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns2:SubmitResponse xmlns:ns3="http://com.vedaadvantage/dp3/Enterprise/System1"
xmlns:ns2="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService"/>
  </soap:Body>
</soap:Envelope>
```

The next example shows almost the same request, but this one is invoking the SubmitWait Web Method.

```
<soapenv:Envelope xmlns:sam="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sys="http://com.vedaadvantage/dp3/Enterprise/System1">
  <soapenv:Header>
<wsse:Security soapenv:mustUnderstand="1" xmlns:wsse="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
<wsu:Timestamp wsu:Id="Timestamp-10" xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
   <wsu:Created>2012-04-12T07:16:35.068Z</wsu:Created>
   <wsu:Expires>2012-04-12T07:17:35.068Z</wsu:Expires>
</wsu:Timestamp>
<wsse:UsernameToken wsu:Id="UsernameToken-9" xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
          <wsse:Username>john</wsse:Username>
           <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
                    username-token-profile-1.0#PasswordText">sfsdfsdfsfsfs</wsse:Password>
         <wsse:Nonce EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
soap-message-security-1.0#Base64Binary">w0eqr+OjrV4OcJLYfWPOWQ==</wsse:Nonce>
         <wsu:Created>2012-04-12T07:16:35.068Z </wsu:Created>
</wsse:UsernameToken>
</wsse:Security>
</soapenv:Header>
  <soapenv:Body>
    <sam:SubmitWait>
      <sys:Transaction>
        <TransactionId>12345</TransactionId>
        <PrimaryApplicant>
          <Name>John</Name>
          <FamilyName>Citizen</FamilyName>
          <State>VIC</State>
          <Age>34</Age>
          <Income>120000</Income>
        </PrimaryApplicant>
      </sys:Transaction>
    </sam:SubmitWait>
  </soapenv:Body>
</soapenv:Envelope>
```

A successful SubmitWait request will provide a response similar to the following:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns2:SubmitWaitResponse xmlns:ns3="http://com.vedaadvantage/dp3/Enterprise/System1"
xmlns:ns2="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService">
      <ns3:Transaction>
        <TransactionId>987654321</TransactionId>
        <PrimaryApplicant>
          <Name>John</Name>
          <FamilyName>Citizen</FamilyName>
          <State>VIC</State>
          <Age>34</Age>
          <Income>120000</Income>
        </PrimaryApplicant>
      </ns3:Transaction>
    </ns2:SubmitWaitResponse>
  </soap:Body>
</soap:Envelope>
```

st

Note:
The following sections will show the remaining examples of valid Business Service requests and their responses. For simplification purposes, the SOAP Header has been left empty (but the security elements are still required when invoking these services).

4.4.2 Retrieve

Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sam="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService"
xmlns:sys="http://com.vedaadvantage/dp3/Enterprise/System1">
  <soapenv:Header/>
  <soapenv:Body>
    <sam:retrieve>
      <sys:TransactionId>2</sys:TransactionId>
    </sam:retrieve>
  </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns2:retrieveResponse xmlns:ns3="http://com.vedaadvantage/dp3/Enterprise/System1"
xmlns:ns2="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService">
      <Transaction>
        <TransactionId>111</TransactionId>
        <PrimaryApplicant>
          <Name>John</Name>
          <FamilyName>Citizen</FamilyName>
          <State>NSW</State>
          <Age>44</Age>
          <Income>70000</Income>
        </PrimaryApplicant>
      </Transaction>
    </ns2:retrieveResponse>
  </soap:Body>
</soap:Envelope>
```

4.4.3 Save

The save request also takes the path-designer defined input type as the input parameter. This allows a transaction that is by design tied to this particular Business Service, to be saved to the BSM persistence layer.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sam="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService"
xmlns:sys="http://com.vedaadvantage/dp3/Enterprise/System1">
  <soapenv:Header/>
  <soapenv:Body>
    <sam:save>
      <sys:Transaction>
        <TransactionId>34678999</TransactionId>
        <PrimaryApplicant>
          <Name>John</Name>
          <FamilyName>Citizen</FamilyName>
          <State>VIC</State>
          <Age>55</Age>
          <Income>12000</Income>
        </PrimaryApplicant>
      </sys:Transaction>
    </sam:save>
  </soapenv:Body>
</soapenv:Envelope>
```

In this case, calling a retrieve with TransactionId **34678999** would return us the transaction we just saved**.**

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns2:saveResponse xmlns:ns3="http://com.vedaadvantage/dp3/Enterprise/System1"
xmlns:ns2="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService"/>
  </soap:Body>
</soap:Envelope>
```

4.4.4 GenerateId

The GenerateId request is identical throughout all Business Services, with the only differing element being the request namespace.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sam="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService">
  <soapenv:Header/>
  <soapenv:Body>
    <sam:generateId/>
  </soapenv:Body>
</soapenv:Envelope>
```

A successful request will provide a response similar to the following:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns2:generateIdResponse xmlns:ns3="http://com.vedaadvantage/dp3/Enterprise/System1"
xmlns:ns2="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService">
      <generatedId>34</generatedId>
    </ns2:generateIdResponse>
  </soap:Body>
</soap:Envelope>
```

4.4.5 GetCacheEntryList

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:con="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService"
xmlns:sys="http://com.vedaadvantage/dp3/Enterprise/System1">
  <soapenv:Header/>
  <soapenv:Body>
    <con:getCacheEntryList>
      <con:transactionId>34567890</con:transactionId>
    </con:getCacheEntryList>
  </soapenv:Body>
</soapenv:Envelope>
```

The response for GetCacheEntryList is a list of CacheEntry elements with each one having an id and name. The id is important for later use in the GetCacheEntry service.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns2:getCacheEntryListResponse xmlns:ns5="http://com.vedaadvantage/dp3/Enterprise/System1"
xmlns:ns4="http://com.vedaadvantage/dp3/connectors/vedaxml/individualcreditscored"
xmlns:ns3="http://com.vedaadvantage/dp3/connectors"
xmlns:ns2="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService">
      <CacheEntryList>
        <cacheEntryId>2250</cacheEntryId>
        <cacheEntryName>vedaxml - individualcreditscored - [ 19/04/2012 11:34:08 ]</cacheEntryName>
      </CacheEntryList>
      <CacheEntryList>
        <cacheEntryId>2250</cacheEntryId>
        <cacheEntryName>vedaxml - individualcreditscored - [ 19/04/2012 11:34:08 ]</cacheEntryName>
      </CacheEntryList>
    </ns2:getCacheEntryListResponse>
  </soap:Body>
</soap:Envelope>
```

4.4.6 GetCacheEntry

As noted in the previous example, the cacheId element in the request of this Web Method is obtained by calling the GetCacheEntryList operation.

```xml
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:con="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService"
xmlns:sys="http://com.vedaadvantage/dp3/Enterprise/System1">
   <soapenv:Header/>
   <soapenv:Body>
      <con:getCacheEntry>
         <con:cacheEntrySearchCriteria>
            <transactionId>34567890</transactionId>
            <cacheId>2250</cacheId>
         </con:cacheEntrySearchCriteria>
      </con:getCacheEntry>
   </soapenv:Body>
</soapenv:Envelope>
```

A valid and successful request will give us the actual report data:

```xml
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns2:getCacheEntryResponse xmlns:ns5="http://com.vedaadvantage/dp3/Enterprise/System1"
xmlns:ns4="http://com.vedaadvantage/dp3/connectors/vedaxml/individualcreditscored"
xmlns:ns3="http://com.vedaadvantage/dp3/connectors"
xmlns:ns2="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService">
         <CacheEntry>
            <reportData><![CDATA[....CacheReportData....]]></reportData>
         </CacheEntry>
      </ns2:getCacheEntryResponse>
   </soap:Body>
</soap:Envelope>
```

4.4.7 SearchMetadata

We have a couple of examples of using the SearchMetadata operation. The first one uses a search expression, based on the metadata defined by the user (the particular details about how this is done are out of the scope of this document);

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:met="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService">
  <soapenv:Header/>
  <soapenv:Body>
    <met:searchMetadata>
      <met:metadataSearchCriteria>
        <expression>Name = 'John Citizen'</expression>
      </met:metadataSearchCriteria>
    </met:searchMetadata>
  </soapenv:Body>
</soapenv:Envelope>
```

The second example performs an transaction search based on core transaction data (createdBy, status, createdOn etc.):

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:met="http://com.vedaadvantage/dp3/Enterprise/System1/SampleSer
vice">
  <soapenv:Header/>
  <soapenv:Body>
    <met:searchMetadata>
      <met:metadataSearchCriteria>
        <applicationCriteria>
          <createdBy>JohnCitizen</createdBy>
          <owner> JohnCitizen </owner>
          <statuses>
            <status>COMPLETED</status>
          </statuses>
        </applicationCriteria>
      </met:metadataSearchCriteria>
    </met:searchMetadata>
  </soapenv:Body>
</soapenv:Envelope>
  </soapenv:Body>
</soapenv:Envelope>
```

The following is an example response for a successful search:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns2:searchMetadataResponse xmlns:ns3="http://com.vedaadvantage/dp3/Enterprise/System1"
xmlns:ns2="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService">
      <MetadataSearchResult>
        <results>
          <result>
            <serviceApplication>
              <id>app_001</id>
              <createdOn>2012-04-19T23:33:54.011Z</createdOn>
              <createdBy> JohnCitizen </createdBy>
              <modifiedOn>2012-04-19T23:33:54.011Z</modifiedOn>
              <modifiedBy> JohnCitizen </modifiedBy>
              <owner> JohnCitizen </owner>
              <status>SAVED</status>
            </serviceApplication>
            <Primary_Applicant_Age>70</Primary_Applicant_Age>
            <Dec>10.1</Dec>
            <Name>John Citizen</Name>
            <Gender>male</Gender>
            <Good>true</Good>
            <Timestamp>2009-05-16T22:42:28Z</Timestamp>
            <BobCreatedAt>2006-08-21Z</BobCreatedAt>
            <BobUpdatedAt>2006-08-21Z</BobUpdatedAt>
          </result>
        </results>
        <totalFound>1</totalFound>
        <pageSize>1</pageSize>
        <pageNumber>1</pageNumber>
      </MetadataSearchResult>
    </ns2:searchMetadataResponse>
  </soap:Body>
</soap:Envelope>
```

5 Troubleshooting

This section will cover some of the more common error scenarios when communicating with BSM services and how to rectify them.

5.1 Failed Authentication

When the request contains invalid user credentials or has no user credentials at all, the service will return a SOAP fault.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <soap:Fault>
        <faultcode>soap:Server</faultcode>
        <faultstring> Service Access Violation</faultstring>
        <detail>
          <ns3:ServiceAccessVerificationFault xmlns:ns2="
http://com.vedaadvantage/dp3/Enterprise/System1/SampleService " xmlns:ns3="
http://com.vedaadvantage/dp3/Enterprise/System1/SampleService1">
            <exception>java.lang.SecurityException<//exception>
           <message> [Failed Authentication : Subject has not been created,
org.springframework.security.authentication.BadCredentialsException: Bad credentials]
</message>
            <dateTime>2012-06-12T01:05:20.826+10:00</dateTime>
            <details>Service Access Violation. </details>
          </ns3:ServiceAccessVerificationFault>
        </detail>
      </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

Solution: Ensure you are using the correct user credentials.

5.2 Failed Authorisation

If the user credentials provided map to a user who doesn’t have the required role to invoke the service, an authorisation failure will follow with the following error:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <soap:Fault>
        <faultcode>soap:Server</faultcode>
        <faultstring> Service Access Violation</faultstring>
        <detail>
           <ns3:ServiceAccessVerificationFault xmlns:ns2="
http://com.vedaadvantage/dp3/Enterprise/System1/SampleService " xmlns:ns3="
http://com.vedaadvantage/dp3/Enterprise/System1/SampleService1">
              <exception>org.springframework.security.access.AccessDeniedException</exception>
             <message>[Access is Denied]</message>
              <dateTime>2012-06-12T01:05:20.826+10:00</dateTime>
              <details>Service Access Violation. </details>
           </ns3:ServiceAccessVerificationFault>
        </detail>
     </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

Solution: Ensure you are using the correct user credentials and/or contact your BSM administrator to assign the particular required role to your user.

5.3 Failed Service Operation Validation

If the transaction is in a state that cannot execute a certain operation, a SOAP fault will be thrown. This applies to a number of scenarios. Some of them are:

- Trying to retrieve a transaction that is submitted for processing and the system haven’t processed yet.
- Executing save on already completed transaction.
- Executing a user defined function when in saved state

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <soap:Fault>
       <faultcode>soap:Server</faultcode>
       <faultstring> Transaction is currently  being processed</faultstring>
       <detail>
         <ns3: ServiceOperationValidatorFault xmlns:ns2="
http://com.vedaadvantage/dp3/Enterprise/System1/SampleService " xmlns:ns3="
http://com.vedaadvantage/dp3/Enterprise/System1/SampleService1">
           <exception> com.vedaadvantage.dp3.manager.servicemanager
.ServiceOperationValidatorFault </exception>
          <message[ Transaction is currently  being processed] </message>
           <dateTime>2012-06-12T01:05:20.826+10:00</dateTime>
           <details>Service Access Violation. </details>
         </ns3: ServiceOperationValidatorFault >
       </detail>
     </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

Solution: Make sure you are executing a web method only when it is in correct state. If polling for completion of a submitted transaction, retry retrieving the transaction after a delay.

5.4 MinInclusive Violation

The Studio designer allows the designer of services to add various restrictions on their request fields, for numbers this includes the **minInclusive** and **maxInclusive** restrictions. Assuming the SampleService SubmitWait request type has a field *Income* with a minimum allowed value of 1000, specifying anything less than that in the request will cause a validation error.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
        <soap:Body>
                <soap:Fault>
                        <faultcode>soap:Client</faultcode>
                        <faultstring>Schema validation failure.</faultstring>
                        <detail>
                                <ns2:SchemaValidationFault xmlns:ns2="
http://com.vedaadvantage/dp3/Enterprise/System1/SampleService ">
                                        <exception>javax.xml.bind.UnmarshalException</exception>
                                        <message> [Unmarshalling Error: cvc-minInclusive-valid: Value '0'
is not facet-valid with respect to minInclusive '1000.0' for type '#AnonType_IncomeApplicant'.]
</message>
                                        <dateTime>2012-06-11T22:22:32.991+10:00</dateTime>
                                        <details>Please correct the schema validation errors. And try again.
</details>
                                </ns2:SchemaValidationFault>
                        </detail>
                </soap:Fault>
        </soap:Body>
</soap:Envelope>
```

Solution: Ensure the value provided in your request is at least of the minimum value allowed in the service WSDL (via the minInclusive attribute)

5.5 MaxInclusive Violation

Here is a sample response when specifying an age greater than the allowed value of 99.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
        <soap:Fault>
                <faultcode>soap:Client</faultcode>
                <faultstring>Schema validation failure.</faultstring>
                <detail>
                        <ns2:SchemaValidationFault xmlns:ns2="
http://com.vedaadvantage/dp3/Enterprise/System1/SampleService ">
                                <exception>javax.xml.bind.UnmarshalException</exception>
                                <message> [Unmarshalling Error: cvc-maxInclusive-valid: Value '500' is not
facet-valid with respect to maxInclusive '99' for type '#AnonType_AgeApplicant'.] </message>
                                <dateTime>2012-06-11T22:22:32.991+10:00</dateTime>
                                <details>Please correct the schema validation errors. And try again. </details>
                        </ns2:SchemaValidationFault>
                </detail>
        </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

Solution: Ensure the value in your request is not greater than the maximum allowed in the service WSDL (via the maxInclusive attribute)

5.6 Invalid Enumeration

Assuming our request has a field of an enumeration type State with only the valid Australian states specified as valid values, anything else provided in the request will cause a schema validation error:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
        <soap:Fault>
                <faultcode>soap:Client</faultcode>
                <faultstring>Schema validation failure.</faultstring>
                <detail>
                        <ns2:SchemaValidationFault xmlns:ns2="
http://com.vedaadvantage/dp3/Enterprise/System1/SampleService ">
                                <exception>javax.xml.bind.UnmarshalException</exception>
                                <message> [Unmarshalling Error: cvc-enumeration-valid: Value 'VICC' is
not facet-valid with respect to enumeration '[VIC, NSW, QLD, WA, TAS, SA, NT]'. It must be a value
from the enumeration.] </message>
                                <dateTime>2012-06-11T22:22:32.991+10:00</dateTime>
                                <details>Please correct the schema validation errors. And try again.</details>
                         </ns2:SchemaValidationFault>
                </detail>
        </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

Solution: Provide a valid value in your request, one from the values listed in the service WSDL.

5.7 Not a Number

Another common schema validation failure is providing a non-numeric value for a field that according to the WSDL is a numeric type. This will cause an error similar to the following:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
        <soap:Fault>
                <faultcode>soap:Client</faultcode>
                <faultstring>Schema validation failure.</faultstring>
                <detail>
                        <ns2:SchemaValidationFault xmlns:ns2="
http://com.vedaadvantage/dp3/Enterprise/System1/SampleService ">
                                <exception>javax.xml.bind.UnmarshalException</exception>
                                <message> [Unmarshalling Error: Not a number: ii] </message>
                                <dateTime>2012-06-11T22:22:32.991+10:00</dateTime>
                                <details>Please correct the schema validation errors. And try again.</details>
                         </ns2:SchemaValidationFault>
                </detail>
        </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

Solution: Ensure all number fields in your request contain valid numeric values

5.8 Number of Elements Violation

Restrictions can also be placed for the minimum and maximum number of elements allowed in a list of elements. Violating this restriction in the request will cause an error similar to the following (here the request had more “OtherApplicant” elements than allowed):

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
         <soap:Fault>
                   <faultcode>soap:Client</faultcode>
                   <faultstring>Schema validation failure.</faultstring>
                   <detail>
                            <ns2:SchemaValidationFault xmlns:ns2="
http://com.vedaadvantage/dp3/Enterprise/System1/SampleService ">
                                     <exception>javax.xml.bind.UnmarshalException</exception>
                                     <message> [Unmarshalling Error: cvc-complex-type.2.4.d: Invalid content
was found starting with element 'sys:Transaction'. No child element '{OtherApplicants}' is expected at
this point.] </message>
                                     <dateTime>2012-06-11T22:22:32.991+10:00</dateTime>
                                     <details>Please correct the schema validation errors. And try again.</details>
                            </ns2:SchemaValidationFault>
                   </detail>
         </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

Solution: Ensure the number of list elements in your request is within the range specified in the WSDL (via the minOccurs and maxOccurs attributes).

5.9 Missing Mandatory Field

If a field is mandatory according to the service WSDL but omitted in the request, a schema validation error will follow:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
         <soap:Fault>
                  <faultcode>soap:Client</faultcode>
                  <faultstring>Schema validation failure.</faultstring>
                  <detail>
                           <ns2:SchemaValidationFault xmlns:ns2="
http://com.vedaadvantage/dp3/Enterprise/System1/SampleService ">
                                    <exception>javax.xml.bind.UnmarshalException</exception>
                                    <message> [Unmarshalling Error: cvc-complex-type.2.4.a: Invalid content was
found starting with element 'FamilyName'. One of '{Name}' is expected. ] </message>
                                    <dateTime>2012-06-11T22:22:32.991+10:00</dateTime>
                                    <details>Please correct the schema validation errors. And try again. </details>
                            </ns2:SchemaValidationFault>
                  </detail>
         </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

Solution: Ensure the missing field is included in your request. Mandatory fields are those with their minOccurs attribute set with a value greater than zero.

5.10 Invalid Namespace

If the namespaces provided in the request do not adhere to the namespaces described in the WSDL, a validation error will occur. For example, if in our request we slightly modify the namespace for the request data;

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sam="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService"
xmlns:sys="http://com.vedaadvantage/dp3/Enterprise/System2">
  <soapenv:Header/>
  <soapenv:Body>
    <sam:Submit>
      <sys:Transaction>
        <TransactionId>346814</TransactionId>
        <PrimaryApplicant>
          <Name>John</Name>
          <FamilyName>Citizen</FamilyName>
          <State>VIC</State>
          <Age>55</Age>
          <!--Optional:-->
          <Income>12000</Income>
        </PrimaryApplicant>
      </sys:Transaction>
    </sam:Submit>
  </soapenv:Body>
</soapenv:Envelope>
```

The following error occurs:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <soap:Fault>
         <faultcode>soap:Client</faultcode>
         <faultstring>Schema validation failure.</faultstring>
         <detail>
                  <ns2:SchemaValidationFault xmlns:ns2="
http://com.vedaadvantage/dp3/Enterprise/System1/SampleService ">
          <exception>javax.xml.bind.UnmarshalException</exception>
          <message> [Unmarshalling Error: cvc-complex-type.2.4.a: Invalid
content was found starting with element 'sys:Transaction'. One of
'{"http://com.vedaadvantage/dp3/Enterprise/System1":Transaction}' is
expected.] </message>
          <dateTime>2012-06-11T22:22:32.991+10:00</dateTime>
          <details>Please correct the schema validation errors. And try again.
</details>
        </ns2:SchemaValidationFault>
      </detail>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

Solution: Ensure you are using the correct namespace; this information will be available from the service WSDL.

5.11 Runtime Fault

If processing the client's request causes a runtime error on the server, a **ServiceExecutionFault** will be returned to the client. The following sample gives an example of a **NullPointerException** encountered during execution:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <soap:Fault>
      <faultcode>soap:Server</faultcode>
      <faultstring>DerivedDataObject:DeriveData encountered an error</faultstring>
      <detail>
        <ns2:ServiceExecutionFault
xmlns:ns2="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService"
xmlns:ns3="http://com.vedaadvantage/dp3/Enterprise/System1">

<exception>com.vedaadvantage.dp3.manager.scriptfunctions.ScriptNullParameterException</exception>
          <message>[Encountered illegal null parameter in script execution]</message>
          <dateTime>2012-04-05T16:19:24.036+10:00</dateTime>
          <details>Please contact the Administrator</details>
        </ns2:ServiceExecutionFault>
      </detail>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

Solution: Contact the BSM administrator and determine the cause of the error.

5.12 SOAP Mismatch

As described earlier in this document, the BSM supports SOAP 1.1. Sending messages in other SOAP formats will cause an error.

Sending a request in SOAP 1.2 format is as easy as changing the namespace, as shown in the following request:

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-
envelope"
xmlns:sam="http://com.vedaadvantage/dp3/Enterprise/System1/SampleService"
xmlns:sys="http://com.vedaadvantage/dp3/Enterprise/System1">
  <soapenv:Header/>
  <soapenv:Body>
    <sam:save>
       <sys:Transaction>
         <TransactionId>346789</TransactionId>
         <PrimaryApplicant>
           <Name>John</Name>
           <FamilyName>Citizen</FamilyName>
           <State>VIC</State>
           <Age>55</Age>
           <!--Optional:-->
           <Income>12000</Income>
         </PrimaryApplicant>
      </sys:Transaction>
    </sam:save>
  </soapenv:Body>
</soapenv:Envelope>
```

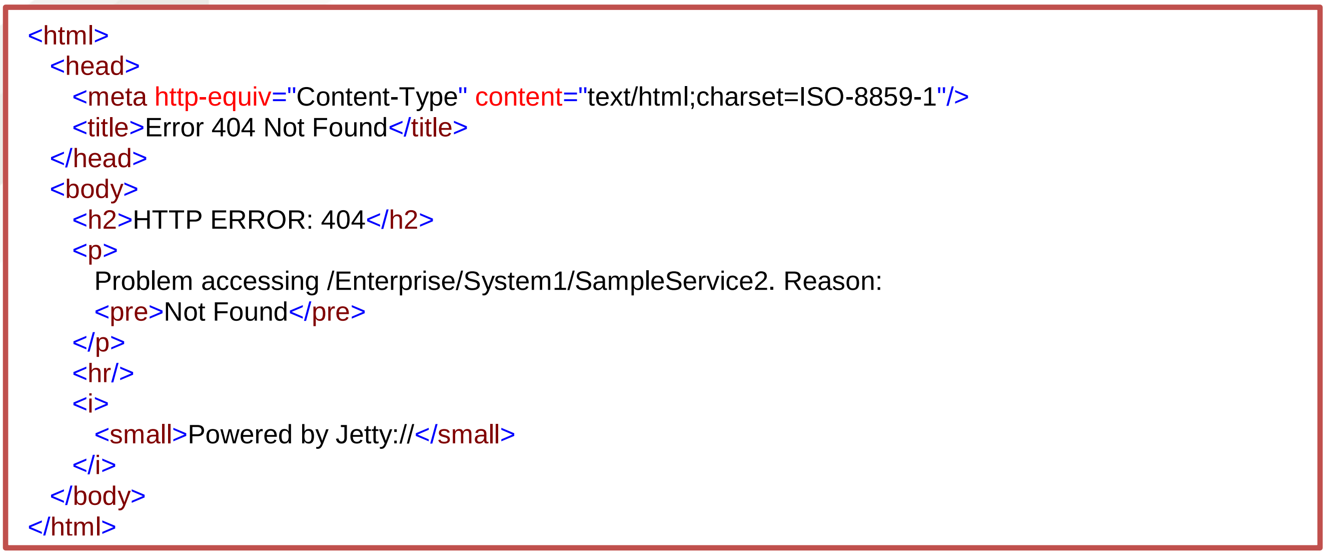
This will cause an error from the server, similar to the following:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
  <soap:Body>
    <soap:Fault>
      <faultcode
xmlns:ns1="http://schemas.xmlsoap.org/soap/envelope/">ns1:VersionMismatch</faultco
de>
      <faultstring>A SOAP 1.2 message is not valid when sent to a SOAP 1.1 only
endpoint.</faultstring>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

Solution: Use the SOAP 1.1 namespace (http://schemas.xmlsoap.org/soap/envelope/) and resubmit the request.

5.13 Invalid Endpoint

If a request is sent to the server with the endpoint to the service being incorrect, an error will be returned. For example, if we slightly modify the endpoint to our SampleService to point to a non-existing SampleService2, the following response will occur:

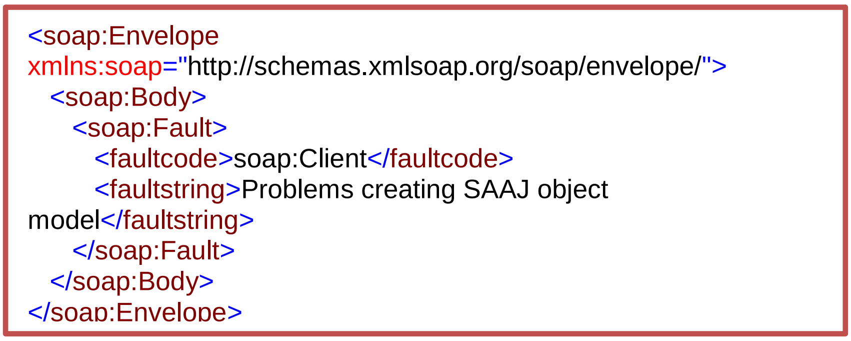


```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html;charset=ISO-8859-1"/>
    <title>Error 404 Not Found</title>
  </head>
  <body>
    <h2>HTTP ERROR: 404</h2>
    <p>
      Problem accessing /Enterprise/System1/SampleService2. Reason:
      <pre>Not Found</pre>
    </p>
    <hr/>
    <i>
      <small>Powered by Jetty://</small>
    </i>
  </body>
</html>
```

Solution: Ensure you are pointing to the correct service endpoint.

5.14 Malformed XML

If the request contains invalid XML the following error will occur:



```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <soap:Fault>
      <faultcode>soap:Client</faultcode>
      <faultstring>Problems creating SAAJ object
model</faultstring>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

Solution: Ensure your request contains valid XML.