ARTICLE

A CASE REPORT OF THE INTERFACE BETWEEN CA-PPM AND SAP VIA WEB-SERVICES

Sriram Nandiraju*

Principal Software Developer, Sabre Inc.USA

ISSN 0976-3104

ABSTRACT

SOAP web services are a powerful way of sending messages over the internet. CA PPM represents a single platform that enables you to manager your entire innovation life cycle and make more informed strategic investments. SAP's ERP system enables companies to run their business processes, be they accounting, sales, production, human resources or payment, in an integrated environment. This case report demonstrates a real-time integration between CA PPM and SAP via SOAP based web services. This case report describes the interface between CA PPM and SAP via web-services for exchanging data through soap requests and getting a response back from SAP.

KEY WORDS
CA PPM, SAP, SOAP, web-service

INTRODUCTION

SOAP [1] provides the envelope for sending Web Services messages over the Internet. It is part of the set of standards specified by the W3C. SOAP is an alternative to Representational State Transfer (REST) and JavaScript Object Notation (JSON). The SOAP envelope contains two parts: 1. An optional header providing information on authentication, encoding of data, or how a recipient of a SOAP message should process the message. 2. The body that contains the message. These messages can be defined using the WSDL specification. SOAP commonly uses HTTP, but other protocols such as Simple Mail Transfer Protocol (SMTP) may be used. SOAP can be used to exchange complete documents or to call a remote procedure.

CA PPM [2] represents a single platform that enables you to manage your entire innovation lifecycle and make more informed strategic investments. CA PPM helps you track and prioritize market and customer requirements and make better decisions on how to invest limited resources, so you can optimize your enterprise, IT, service and product portfolio.

In this case study, we shall look at a use case wherein CAPPM will send a SOAP payload to SAP for creating a project definition and a separate SOAP payload to create task level (WBS level) structure of the project in SAP.

The programming interface in CA PPM is known as gel-scripting and it has the capability to query the data from the CA PPM database and send that data over to any third-party endpoint via a SOAP message (in XML format).

Received: 22 Oct 2017
Accepted: 27 Dec 2017
Published: 15 Jan 2018

There are very few studies demonstrating the communication between CAPPM and SAP via web services. This study demonstrates the real-time integration between CAPPM and SAP via web services.

GEL SCRIPT in CAPPM

The below script sends a project from CAPPM to SAP.

```
<gel:script
xmlns:core="jelly:core",xmlns:file="jelly:com.niku.union.gel.FileTagLibrary"
xmlns:gel="jelly:com.niku.union.gel.GELTagLibrary"  xmlns:nikuq="http://www.niku.com/xog/Query"
   xmlns:soap="jelly:com.niku.union.gel.SOAPTagLibrary"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"  xmlns:sql="jelly:sql"  xmlns:util="jelly:util"
   xmlns:x="jelly:org.apache.commons.jelly.tags.xml.XMLTagLibrary"
xmlns:xog="http://www.niku.com/xog"  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <gel:log>Start this Script</gel:log>
   <!-- Set the endpoint to ServiceNow and set our Authorization for Basic -->
   <core:set
 var="soapEndPoint"
value="https://integration.cert.<yourcompany>.com/XISOAPAdapter/MessageServlet?senderParty=&amp;
senderService=ESERVICES_Q&amp;receiverParty=&amp;receiverService=&amp;interface=si_os_VIB_HOS
T_CIRCUIT&amp;interfaceNamespace=http://EservicesBapi2Webservice.com" />  ☐ specify the WSDL's
end-point here. This was provided by SAP
As demonstrated by Gene. [3]
<core:invokeStatic  var="base64"  className="com.niku.union.utility.Base64"  method="encode">  ☐
readying for Base64 authentication
      <core:arg  type="java.lang.String"  value="soap_test:XXXXXXXX" />  ☐ specify the user-name and
password. This was provided by SAP
   </core:invokeStatic>
   <core:set var="basicAuth" value="Basic ${base64}" />
   <gel:log>basicAuth = ${basicAuth}</gel:log>
```

***Corresponding Author**
Email:
clarity.ram@gmail.com
Tel.: +1-312 970 9116

IIOAB JOURNAL
ISSN: 0976-3104

```
<!-- Open a connection to ServiceNow and set our request headers -->
<core:new var="soapUrl" className="java.net.URL">
  <core:arg type="java.lang.String" value="${soapEndPoint}" />
</core:new>
<core:invoke var="connection" on="${soapUrl}" method="openConnection"/>
<core:expr value="${connection.setDoOutput(true)}" />
<core:expr value='${connection.setRequestMethod("POST")}'/>
<core:expr value='${connection.setRequestProperty("Content-type", "text/xml; charset=utf-8")}'/>
<core:expr value='${connection.setRequestProperty("SOAPAction", soapEndPoint)}'/>
<core:expr value='${connection.setRequestProperty("Authorization", basicAuth)}'/> ☐ passing the
Base64 authentication
<core:set var="requestXml">
        <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:urn="urn:sap-com:document:sap:rfc:functions">
          <soapenv:Header/>
          <soapenv:Body>
                  <urn:Z_BAPI_VIB_HOST_CIRCUIT> ☐ This is the soap body which is defined through
the wsdl. We can get the soap-body from a tool like SOAP UI
                        <CUSTOMER_NO>xWD40</CUSTOMER_NO>
                        <ZBAPIEVCIRC/>
                        <ZBAPIEVHOST/>
                  </urn:Z_BAPI_VIB_HOST_CIRCUIT>
          </soapenv:Body>
        </soapenv:Envelope>
</core:set>
<!-- Write out our getRecords request to ServiceNow -->
<core:invoke var="outputStream" on="${connection}" method="getOutputStream" /> ☐ invoking the
web-service
<core:new var="outputStreamWriter" className="java.io.OutputStreamWriter">
  <core:arg type="java.io.OutputStream" value="${outputStream}" />
</core:new>
<core:expr value="${outputStreamWriter.write(requestXml)}" />    <core:expr
value="${outputStreamWriter.close()}" />
<!-- Read in the response from ServiceNow into a string -->
<core:invoke var="inputStream" on="${connection}" method="getInputStream" /> ☐ getting the
response back
<core:new var="inputStreamReader" className="java.io.InputStreamReader">
    <core:arg type="java.io.InputStream" value="${inputStream}" />
</core:new>
<core:new var="stringBuilder" className="java.lang.StringBuilder" />
<core:set var="data" value="${inputStreamReader.read()}" />
<core:while test="${data != -1}">
<core:invokeStatic var="char" className="java.lang.Character" method="toChars" >
  <core:arg type="int" value="${data}" />
</core:invokeStatic>
<core:invokeStatic var="charString" className="java.lang.String" method="valueOf" >
  <core:arg value="${char}" />
</core:invokeStatic>
  <core:set var="char" value="${java.lang.String.valueOf(java.lang.Character.toChars(data))}"/>
    <core:invoke method="append" on="${stringBuilder}">
      <core:arg value="${charString}" />
    </core:invoke>
    <core:set var="data" value="${inputStreamReader.read()}" />
</core:while>
<!-- Here is a string of the xml reponse payload -->
<gel:log>${stringBuilder.toString()}</gel:log>      <gel:log>End this Script</gel:log>   </gel:script>
```

RESULTS

Below are the results from the SOAP web service call made from CA PPM to SAP. Tests are done with correct and incorrect payload, incorrect password to demonstrate the accuracy of the tests that were performed {Table 1].

Table 1: Test results from the SOAP web service call made from CA PPM to SAP

Step Name	Messages from CA PPM	Comments from Ram
correct pwd correct parameters	Start this Script	Start of the web-service with valid customer data
correct pwd	basicAuth = Basic c29hcF90ZXN0OmhvY2tleTE2	Correct

IIOAB JOURNAL
ISSN: 0976-3104

correct parameters		Authentication information was provided to the web-service from inside the CA PPM script.
correct pwd correct parameters	<SOAP:Envelope xmlns:SOAP='http://schemas.xmlsoap.org/soap/envelope/'><SOAP:Header/><SOAP:Body><n0:Z_BAPI_VIB_HOST_CIRCUIT.Response xmlns:n0='urn:sap-com:document:sap:rfc:functions' xmlns:prx='urn:sap.com:proxy:SEP:/1SAI/TAS226DF51D1D340203BB9B:740'><HSTCNT>6</HSTCNT><ZBAPIEVCIRC/><ZBAPIEVHOST><item><KUNNR>XWD40</KUNNR><HOST_KEY>0000024753</HOST_KEY><PROD_LINE>S</PROD_LINE><CNCT_PROF>07</CNCT_PROF><CNTRACT_NO>2030079511</CNTRACT_NO><RCUR_BILL>2030079511</RCUR_BILL><CTRT_CURR>USD</CTRT_CURR><GRPINGID>59SF</GRPINGID><LEVEL_TA>HIGH/6</LEVEL_TA><TCP_IP>Y</TCP_IP><CRT_CNT>0024</CRT_CNT><PRIME_LNK>N</PRIME_LNK><SAP_P_LNNO>000000</SAP_P_LNNO><VIB_STAT>IN</VIB_STAT><EID>0000024753</EID><CREATEDATE>2015-05-06</CREATEDATE></item><item><KUNNR>XWD40</KUNNR><HOST_KEY>0000104041</HOST_KEY><PROD_LINE>S</PROD_LINE><CNCT_PROF>07</CNCT_PROF><CNTRACT_NO>2030079511</CNTRACT_NO><RCUR_BILL>2030079511</RCUR_BILL><CTRT_CURR>USD</CTRT_CURR><GRPINGID>5HL9</GRPINGID><LEVEL_TA>HIGH/6</LEVEL_TA><TCP_IP>Y</TCP_IP><CRT_CNT>0027</CRT_CNT><PRIME_LNK>N</PRIME_LNK><SAP_P_LNNO>000000</SAP_P_LNNO><VIB_STAT>IN</VIB_STAT><EID>0000104041</EID><CREATEDATE>2015-05-06</CREATEDATE></item><item><KUNNR>XWD40</KUNNR><HOST_KEY>0000154305</HOST_KEY><PROD_LINE>S</PROD_LINE><CNCT_PROF>07</CNCT_PROF><CNTRACT_NO>2030079511</CNTRACT_NO><RCUR_BILL>2030079511</RCUR_BILL><CTRT_CURR>USD</CTRT_CURR><GRPINGID>CAQ8</GRPINGID><LEVEL_TA>HIGH/6</LEVEL_TA><TCP_IP>Y</TCP_IP><CRT_CNT>0026</CRT_CNT><PRIME_LNK>N</PRIME_LNK><SAP_P_LNNO>000000</SAP_P_LNNO><VIB_STAT>IN</VIB_STAT><EID>0000154305</EID><CREATEDATE>2015-05-06</CREATEDATE></item><item><KUNNR>XWD40</KUNNR><HOST_KEY>0000114233</HOST_KEY><PROD_LINE>S</PROD_LINE><CNCT_PROF>07</CNCT_PROF><CNTRACT_NO>2030079511</CNTRACT_NO><RCUR_BILL>2030079511</RCUR_BILL><CTRT_CURR>USD</CTRT_CURR><GRPINGID>CN41</GRPINGID><LEVEL_TA>HIGH/6</LEVEL_TA><TCP_IP>Y</TCP_IP><CRT_CNT>0029</CRT_CNT><PRIME_LNK>N</PRIME_LNK><SAP_P_LNNO>000000</SAP_P_LNNO><VIB_STAT>IN</VIB_STAT><EID>0000114233</EID><CREATEDATE>2015-05-06</CREATEDATE></item><item><KUNNR>XWD40</KUNNR><HOST_KEY>0000114232</HOST_KEY><PROD_LINE>S</PROD_LINE><CNCT_PROF>07</CNCT_PROF><CNTRACT_NO>2030079511</CNTRACT_NO><RCUR_BILL>2030079511</RCUR_BILL><CTRT_CURR>USD</CTRT_CURR><GRPINGID>CN4F</GRPINGID><LEVEL_TA>HIGH/6</LEVEL_TA><TCP_IP>Y</TCP_IP><CRT_CNT>0033</CRT_CNT><PRIME_LNK>N</PRIME_LNK><SAP_P_LNNO>000000</SAP_P_LNNO><VIB_STAT>IN</VIB_STAT><EID>0000114232</EID><CREATEDATE>2015-05-06</CREATEDATE></item><item><KUNNR>XWD40</KUNNR><HOST_KEY>0000114132</HOST_KEY><PROD_LINE>S</PROD_LINE><CNCT_PROF>07</CNCT_PROF><CNTRACT_NO>2030079511</CNTRACT_NO><RCUR_BILL>2030079511</RCUR_BILL><CTRT_CURR>USD</CTRT_CURR><GRPINGID>CND1</GRPINGID><LEVEL_TA>HIGH/6</LEVEL_TA><TCP_IP>Y</TCP_IP><CRT_CNT>0031</CRT_CNT><PRIME_LNK>N</PRIME_LNK><SAP_P_LNNO>000000</SAP_P_LNNO><VIB_STAT>IN</VIB_STAT><EID>0000114132</EID><CREATEDATE>2015-05-06</CREATEDATE></item></ZBAPIEVHOST></n0:Z_BAPI_VIB_HOST_CIRCUIT.Response></SOAP:Body></SOAP:Envelope>	Call made successfully with the correct data items and the web-service from SAP has sent the response with the customer data.
correct pwd **Incorrect** parameters	Start this Script	Start of the web-service with invalid customer data
correct pwd **Incorrect** parameters	basicAuth = Basic c29hcF90ZXN0OmhvY2tleTE2	Correct Authentication information was provided to the web-service from inside the CA PPM script.
correct pwd	<SOAP:Envelope	Call made

IIOAB JOURNAL
ISSN: 0976-3104

Incorrect parameters	xmlns:SOAP='http://schemas.xmlsoap.org/soap/envelope/'><SOAP:Header/><SOAP:Body><n0:Z_BAPI_VIB_HOST_CIRCUIT.Response xmlns:n0='urn:sap-com:document:sap:rfc:functions' xmlns:prx='urn:sap.com:proxy:SEP:/1SAI/TAS226DF51D1D340203BB9B:740'><RETURN><TYPE>**E**</TYPE><ID>**Z0**</ID><NUMBER>112</NUMBER><MESSAGE>Invalid Customer</MESSAGE><MESSAGE_V1> Customer</MESSAGE_V1><SYSTEM>SEQ010</SYSTEM></RETURN><ZBAPIEVCIRC/><ZBAPIEVHOST/></n0:Z_BAPI_VIB_HOST_CIRCUIT.Response></SOAP:Body></SOAP:Envelope>	successfully with in- correct data items and the web-service from SAP has sent the response with an error message **EZ0**
wrong pwd	BPM-0704: An error occurred while executing custom script: org.apache.commons.jelly.JellyTagException: null:57:81: <core:invoke> method getInputStream threw exception: Server returned **HTTP response code: 401** for URL: https://integration.cert.sabre.com/XISOAPAdapter/MessageServlet?senderParty=&senderService=ESERVICES_Q&receiverParty=&receiverService=&interface=si_os_VIB_HOST_CIRCUIT&interfaceNamespace=http://sabreEservicesBapi2Webservice.com at org.apache.commons.jelly.tags.core.InvokeTag.doTag(InvokeTag.java:109) at org.apache.commons.jelly.impl.TagScript.run(TagScript.java:247) at org.apache.commons.jelly.impl.ScriptBlock.run(ScriptBlock.java:95) at org.apache.commons.jelly.TagSupport.invokeBody(TagSupport.java:186) at com.niku.union.gel.tags.ScriptTag.doTag(ScriptTag.java:20) at org.apache.commons.jelly.impl.TagScript.run(TagScript.java:247) at com.niku.union.gel.GELScript.run(GELScript.java:57) at com.niku.union.gel.GELController.invoke(GELController.java:74)	Start of the web-service with in-correct credentials. Web-service failed with **401** error indicating that it is an authorization issue.

CONCLUSION

The article is relevant because the documentation on web services in CA PPM is lacking, and is a benefit to CAPPM developers as SAP integration is in demand now.

The syntax of jelly script is one of the more difficult syntaxes to read, and the apache documentation is limited. Therefore, having an example like this denoting the needed attributes for class creation, usage, and calling will be helpful.

CONFLICT OF INTEREST
There is no conflict of interest.

ACKNOWLEDGEMENTS
None

FINANCIAL DISCLOSURE
None

REFERENCES

[1] https://www.service-architecture.com/articles/web-services/soap.html
[2] https://ca.com/us/products/ca-project-portfolio-management.html
[3] A post from the user “Gene" with userid “gcubed” dated “Dec 24, 2014 3:54 PM” in the URL https://catech.hosted.jivesoftware.com/thread/241700025”