Web Service Access in Enterprise Portal: Issues and Solution

Applies to:

This article applies to SAP NetWeaver - Enterprise Portal.

Summary

The procedure of accessing a project published as web services in SAP Enterprise Portal is presented. The document focuses on the issues and solution encountered in implementing this procedure. Web services deployed in SAP and non-SAP environment can be utilized in a SAP application which is the first step to adopt SOA architecture.

Author: Manimekalai Mathu

Company: HCL Technologies

Created on: 07 January 2008

Author Bio

Manimekalai Mathu is working as a developer in Engineering and R&D services of HCL Technologies in Chennai, India.

Table of Contents

Introduction

Web services provide a standard means of interoperating between different software applications, running on a variety of platforms and/or frameworks.

SAP NetWeaver is a web-based, open integration and application platform that serves as the foundation for enterprise service-oriented architecture and allows the integration and alignment of people, information, and business processes across business and technology boundaries

The SAP Enterprise Portal enables to access applications from other systems and sources, such as the Internet or intranet using one entry point. Enterprise portal is the basic way of creating the iView that is an end point access.

Accessing a Web Service in a Portal Component through a Portal Service encounters few issues while implementation. The procedure, issues, and solutions are presented below.

Procedure to Access a Web Service in Enterprise Portal

SAP Enterprise Portal has Portal Runtime, which includes Portal Service and Portal Component. The common way of accessing a Portal Service is to publish it as a web service. NetWeaver supports the Portal Service published as web service and hence utilized by external users. An alternate way of implementation gets adopted i.e. using an external web service to create the Portal Service and making it usable in SAP environment.

SAP NetWeaver SP9 - Enterprise Portal 7.0.09 is used. Web service has a wsdl file that is accessed in a Portal Service, has the Interface of the web service. Portal service is created by providing the wsdl file path of the web service. Portal service has the interface with methods of the web service. The web service methods are present in the Interface of Portal Service. The Portal Component has a service call to the Portal Service methods and makes use of the interfaces to communicate with web service.

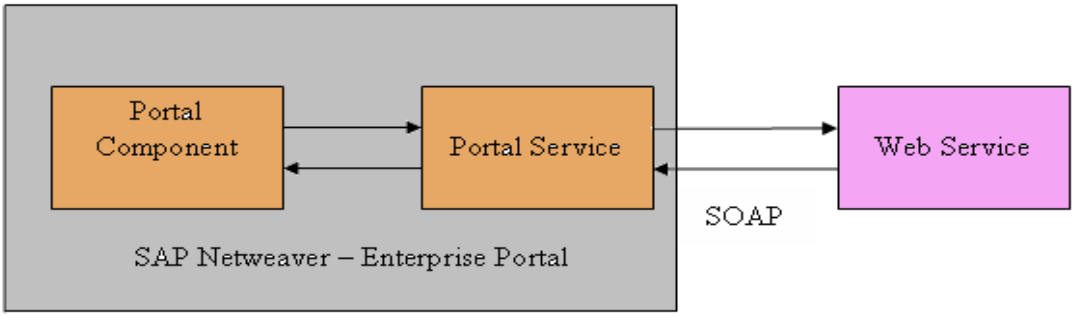

Figure 1: Web Service Access in Enterprise Portal

Developed Project is Published as Web Service

As you see in figure 2, the deployed web services are in the SAP web service navigator.

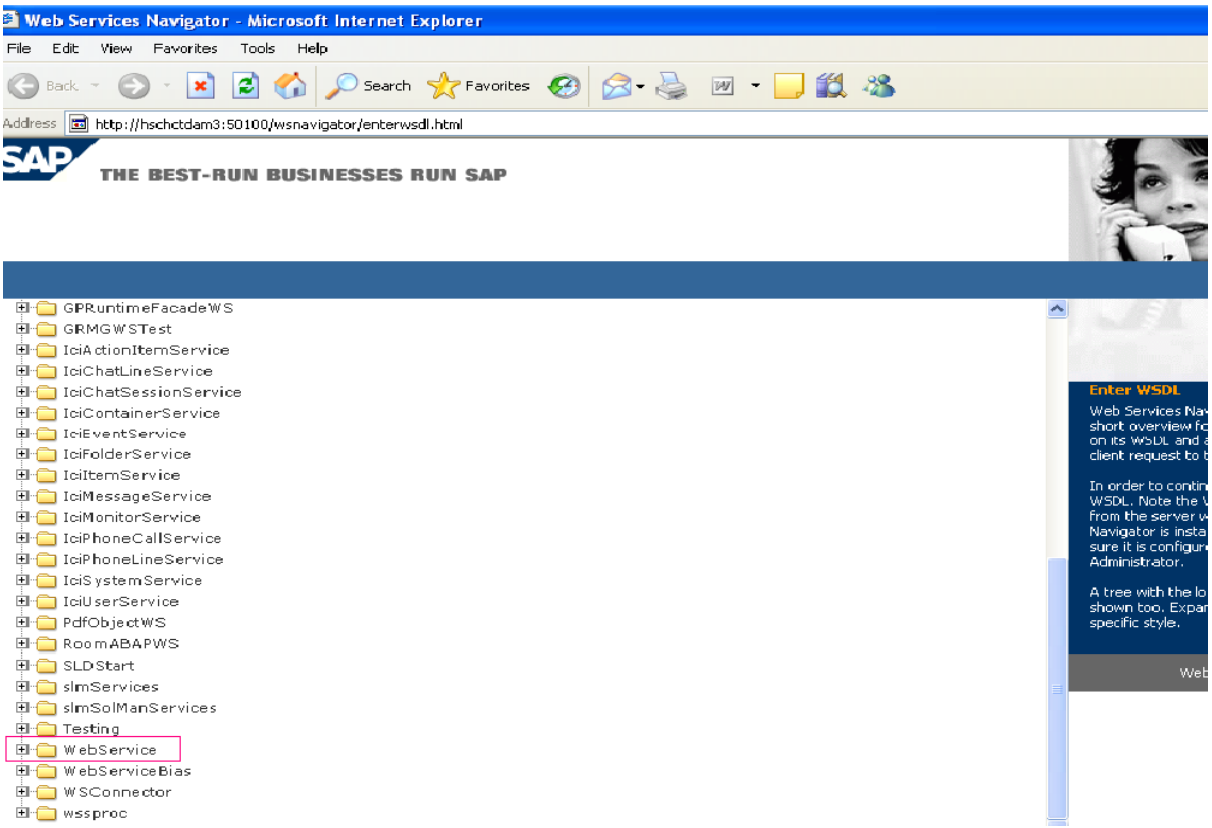

Figure 2: web service deployed in web service navigator

In figure 3, web service is tested by giving the string “SAP” as request and receiving the string “Hello SAP” as response.

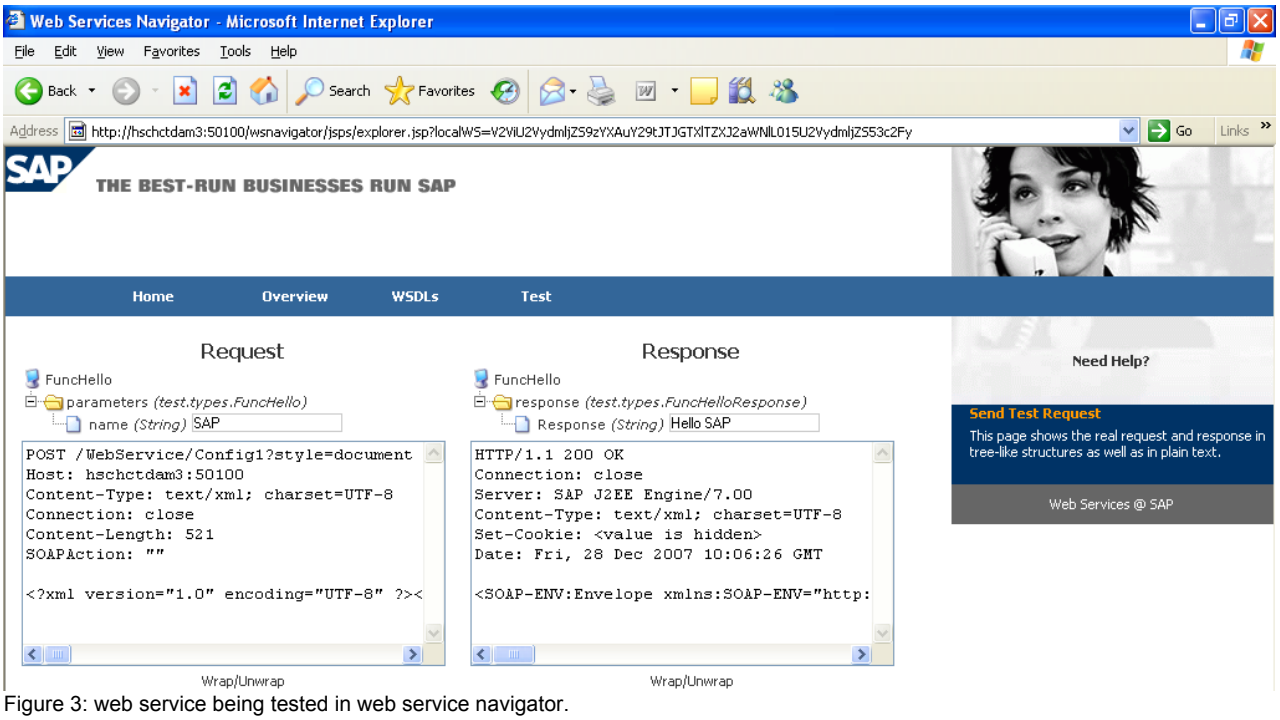

Figure 3: web service being tested in web service navigator.

Portal Service is Created Using the WSDL File of the Web Service

A new project is created and a Portal Service is developed in it, by giving the wsdl path of web service. It contains the Interface that has the methods of the web service.

The "Portal Service from wsdl file - Client side" wizard allows generating a portal service from an external web service's wsdl file. The generated portal service acts as a proxy or client to the external web service or application.

The step by step procedure to create a portal service is as follows:

1. Launch NetWeaver Developer Studio
2. Create a Portal Application Project. To create a portal application project follow the below mentioned steps
3. From the File menu, select New... Other. The New window is displayed.
4. Select Portal Application, and then Create a Portal Application Project. Click Next.
5. In the Project name textbox, enter a name for the project, say Demo1 and location like C:/SAP.
6. Click Finish.
7. Create a PortalService. To create an PortalService follow the below mentioned steps
8. From the File menu, select new → other. The New window is displayed.
9. Select Portal Application, and then Create a New Portal Application Object. Click Next.
10. Select the project to which you want to add the portal service (Demo1Comp.java). Click Next.
11. Select Portal Web Service → Portal Service from Wsdl file – Client side. Click Next.
12. The next wizard, give the path of wsdl file. Click next.
13. The methods in the web service are displayed. Select the methods need to be accessed and click next.

    Enter the following fields:

    ServiceName: Name of portal service (say Demo1)

    Alias: The Alias name for the portal service is needed which is used as a key to access the methods in portal service.

    Package name: The package name for the Java class file for this component (say com.sap.portal.webservice.Demo1).

14. Click Finish.

Figure 4 has the portal service Demo1 with method FuncHello from web service. Portal service does not have the user interface. It can be accessed by using portal component.

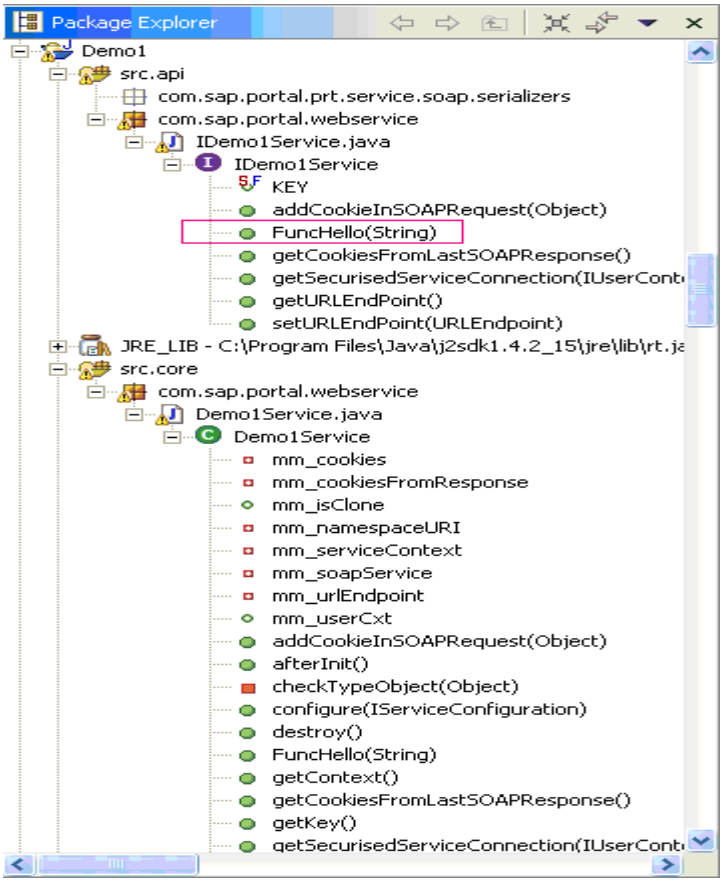

Figure 4: portal service created using wsdl file path

Portal Component is Created and the Methods of the Portal Service are Accessed in it

The component has the User Interface, which makes use of the portal service methods.

1. ¸ Launch NetWeaver Developer Studio
2. Create a Portal Application Project. To create a portal application project follow the below mentioned steps
3. From the File menu, select New... Other. The New window is displayed.
4. Select Portal Application, and then Create a Portal Application Project. Click Next.
5. In the Project name textbox, enter a name for the project, say Demo1Comp.
6. Click Finish.
7. Create an AbstractPortalComponent. To create an AbstractPortalComponent follow the below mentioned steps
8. From the File menu, select new → other. The New window is displayed.
9. Select Portal Application, and then Create a New Portal Application Object. Click Next.
10. Select the project to which you want to add the portal component (Demo1Comp.java). Click Next.

11. Select Portal Component → AbstractPortalComponent. Click Next.

    Enter the following fields:

    Name: Name of component (say Demo1Comp.java)

    Location: The location of the new Java class file for this component (say C:\SAP).

    Class name: The Java classes file for this component.

    Package name: The package name for the Java class file for this component (say com.sap.portal.webservice.Demo1Comp).

12. Click Finish.

Now the portal component is created. The code for Demo1Comp.java is written in it.

Portal Component is Deployed in the Server and Ready to use by End Users

Figure 5 shows the portal page with deployed portal service Demo1 and portal component Demo1Comp.

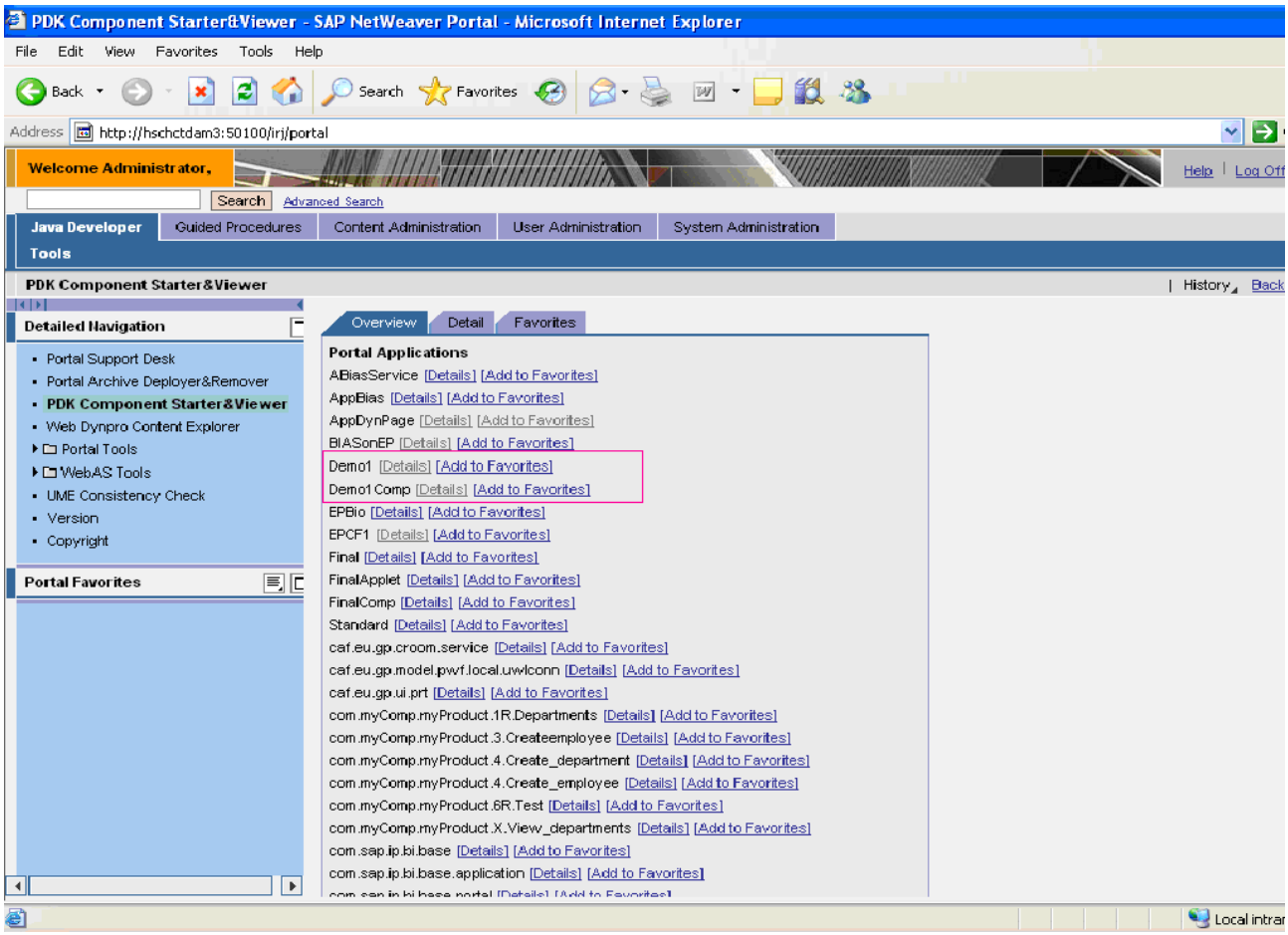

Figure 5: Portal Services and Components deployed in SAP server.

Sample Code

The sample code to access the above service in a portal component is:

WebService.java

```
//The code in web service with Function FuncHello
package com.sap;
public class WebService
{
   public String FuncHello(String name)
   {
          return "Hello "+name;
   }
}
```

Demo1Comp.java

```
//FuncHello method accessed using IDemo1Service Interface of portal service.

package com.sap.portal.webservice;

import com.sapportals.portal.prt.component.*;
import com.sap.portal.webservice.IDemo1Service;
import com.sapportals.portal.prt.runtime.PortalRuntime;
import com.sapportals.portal.prt.component.IPortalComponentRequest;
import com.sapportals.portal.prt.component.IPortalComponentResponse;

public void doContent(IPortalComponentRequest request, IPortalComponentResponse
response)
    {
         String text = “EP”;

return (  (IDemo1Service) PortalRuntime
                                    .getRuntimeResources()
                                    .getService(
               IDemo1Service.KEY)).FuncHello(text);
          }
```

Output:

The output of this component is “Hello EP”.

Issues and Solution

The issues encountered while following the procedure and their corresponding solutions are given below.

1. To create a portal service, the wsdl file path is required to generate the interface for the methods of web service deployed in SAP. In this case, the server, which has the SAP installed, has the wsdl path. In some cases, on giving the path with server hostname, the wsdl is not identified.

   Solution: The preferred way to access the wsdl file path is to download the file in local machine and give that path by clicking the browse button of portal service creation wizard. This will solve many issues in web service access. Figure 6 has the wsdl file path, to create the portal service.

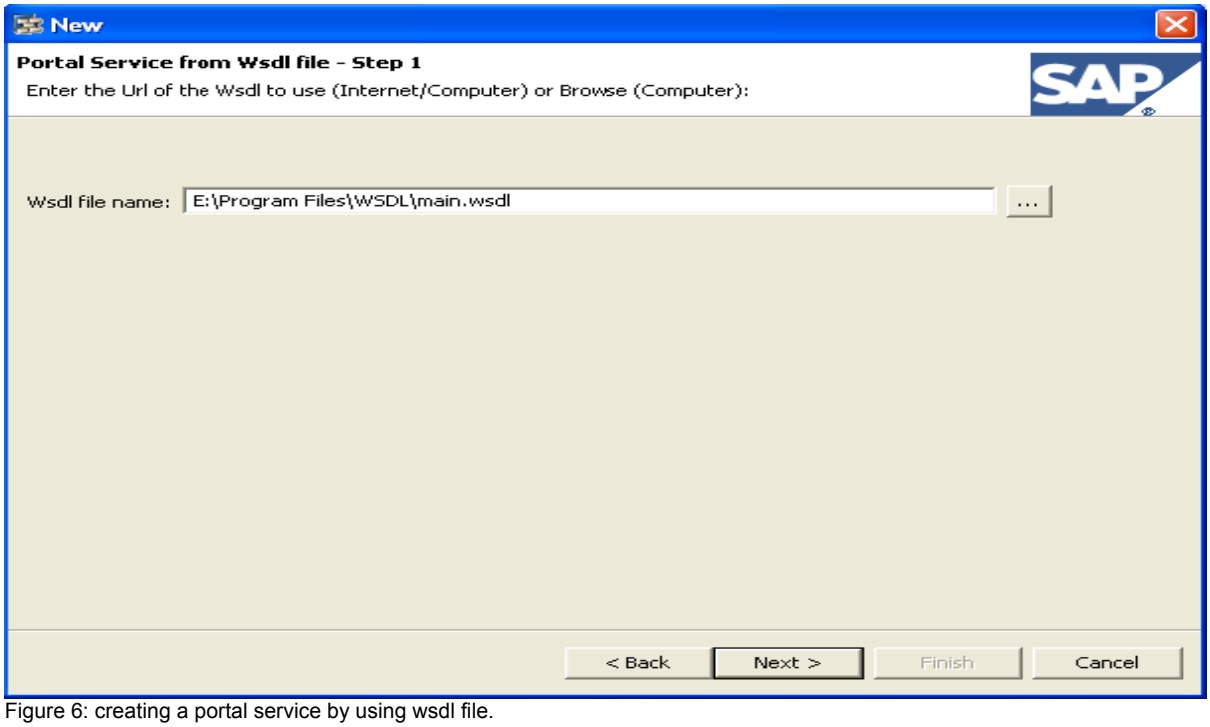

Figure 6: creating a portal service by using wsdl file.

2. While creating Portal Service using the IP address of the server, the methods in the interface have parameter passed as request and responses. There are complexities in accessing request and response in a Portal Component.

   The portal service NewPortal in the figure 7 has the Interface INewPortal which has the method FuncHello with response as parameter.

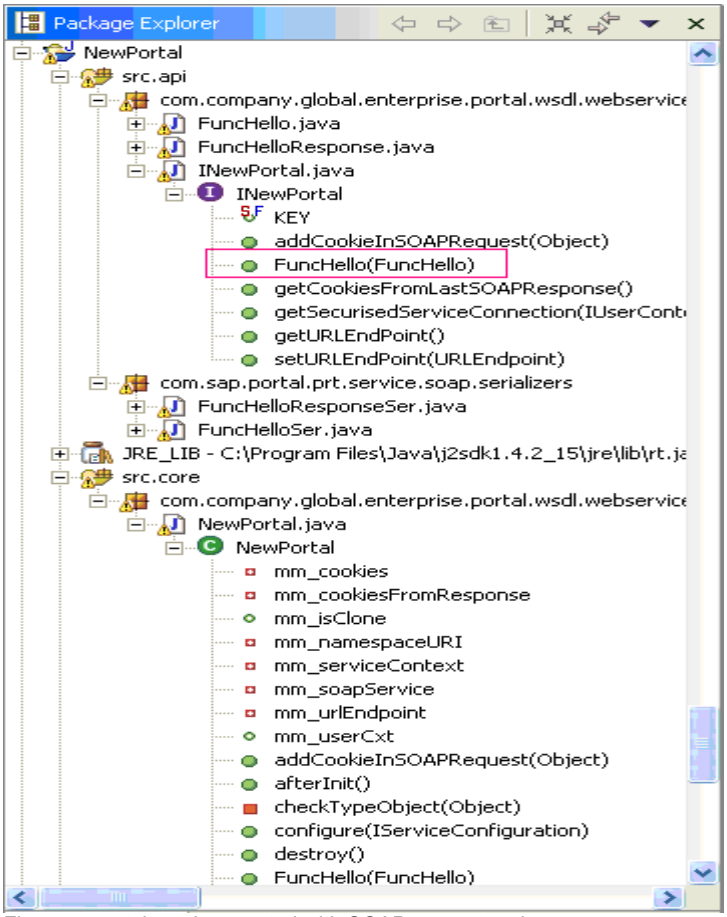

Figure 7: portal service created with SOAP request and response.

Solution: The methods in the Portal Service are request and responses. This is caused by giving the path of wsdl file with hostname or ip address where the web service is deployed. If the wsdl file is downloaded and utilized, the parameters passed reflect the same as the web service.

3. After creating the iView, it is deployed and open the iView, which sometimes throws a Portal Runtime Exception as “iView not found” or “iView NA” (Not Applicable).

   Solution: To solve this issue, Private sharing reference has to be set to link the component with portal service. The figure 8 is the SharingReference property window of portal component which has the portal service name in it.

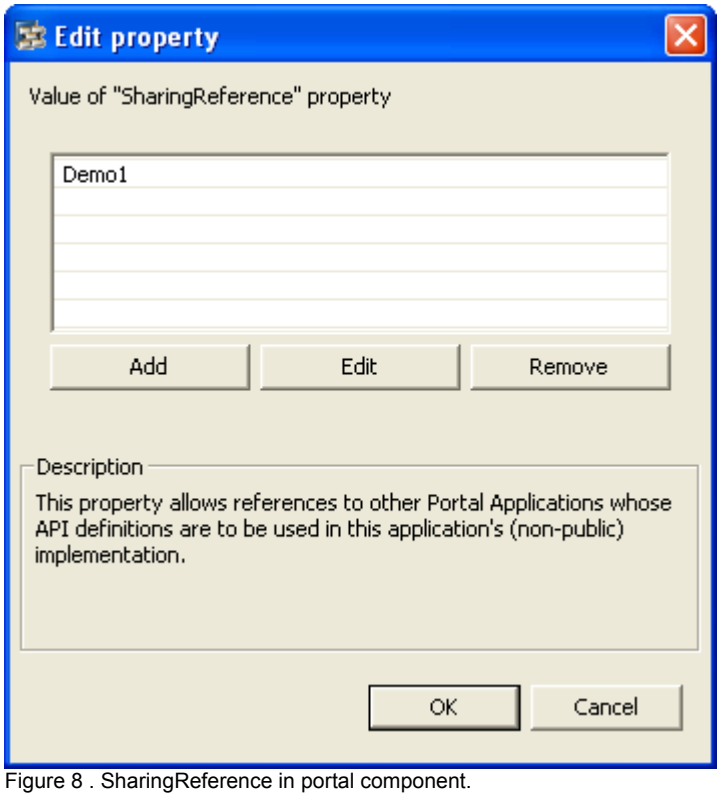

Figure 8 . SharingReference in portal component.

4. Once the iView is created, it is deployed in server and viewed for output which produces an exception SOAP response as null i.e. SOAP response “” as a Portal Runtime Error. It means that the response produced by the PRT is null.

   Solution: The SOAP response “” Portal Runtime Error occurs if SOAP response is null. In this case, the downloaded wsdl file is with SOAP Action “”. Each method in the Interface uses the SOAP Action. All SOAP actions in bindings folder are filled with some string like “MyService”.

5. In case of server being an external machine, authentication problem is encountered since; the response has to be received from the proxy server.

   Solution: The authentication problem is caused due to the lack of proxy settings in SAP server. This problem is brought to an end by giving the proxy details in the server configuration. In Portal window, Service Configuration set the http Service Bypass proxy server to hostname of the SAP server. Figure 9 has the proxy settings which have to be set in SAP server.

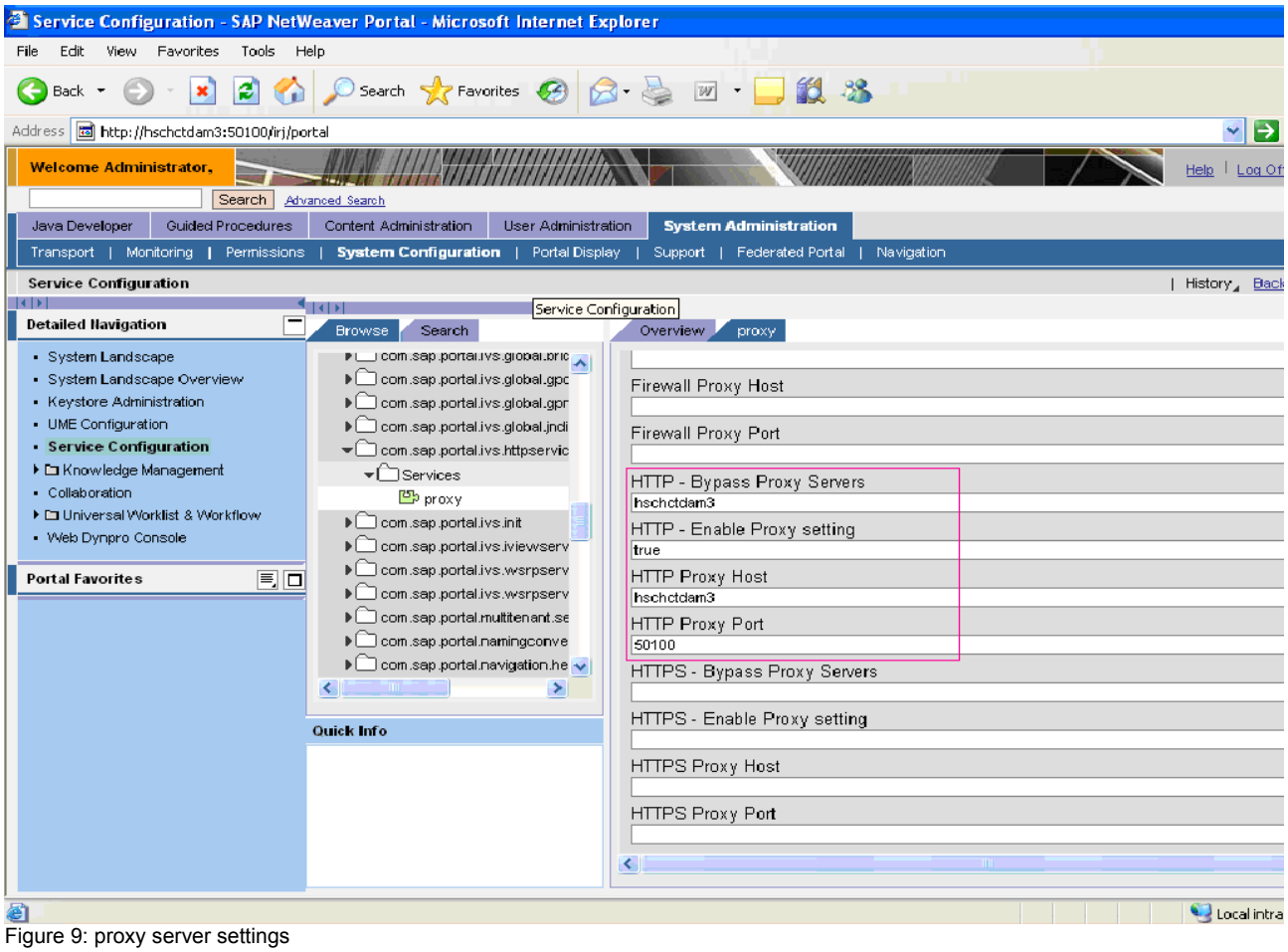

Figure 9: proxy server settings

Conclusion

The advantage of SAP portal is the availability to integrate business packages. These business packages predefined for a business area, provides a significant way in development of vertical business solutions. The procedure involves easy access of Business process. Once the software development is completed, the Business process is exposed as Web Service and made usable by wide range of tools. The Web Service is implemented in SAP NetWeaver and made usable on EP. Implementing on SAP NetWeaver provides easy access to Business methods.

Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.