

Generating Web Service Manually for a Rules Project using Java



Applies to

SAP NetWeaver Composition Environment 7.1.1

Summary

The business rules created using the rules composer of NWDS can be invoked either by coding the invocation logic directly in the application or by exposing the rules as a web service and using this web service in any application as required.

This article will help you in manually generating a web service for a rules project using java class as its business object.

Author: Arti Gopalan

Company: SAP Labs India Pvt. Ltd.

Created on: 2 October 2008

Table of Content

Introduction	3
Creating the DCs	3
Creating Web Service.....	5
Getting the WSDL from the WS Navigator	6
Example	7
Generating Web Service DCs.....	7
Customizing the DCs	7
Copyright.....	8

Introduction

Business rules describe the operations and constraints that are applicable to an organization and thus help in guiding the business processes. NetWeaver allows you to cumulate these sets of business rules by making use of the rules composer DC.

Once the policies of an organization has been captured in the form of business rules using the BRM tool of the NWDS, we now have to decide on how these rules can be used in our application. BRM allows you to either invoke the rule set directly in your application by coding the invocation logic directly in the application or allows you to expose these rules as a web service, which can be later used in any application as required.

Rules can be written using either XML Schema or Java classes as its business object.

This article helps you in generate a web service manually for a rules project using java class as its business object. To do this we make use of the web service generator tool provided with the NWDS.

Creating the DCs

The following are the steps to be performed to create a web service–

1. Open your *NWDS*.
2. Create a new Development Component project of type *J2EE > Web Module*.
3. Create a new Development Component of type *J2EE > Enterprise Application*. Associate it to the web module DC created previously.
4. Add the following dependencies to both the web module DC as well as to the Enterprise Application DC.
 - a. BRMS façade
 - b. Engine façade
 - c. Logging – *tc/bl/logging/api*
 - d. The DC which contains the Java resource used by the Rules project.
5. Create a new Java class in the source folder of the Web Module DC.

An example of the Java class to be created –

```

package com.sap.helper;

public class EngineInvoker {

    private static String jndiName = "com.sap.brms.RuleEngine";
    private static String projectName = "<Project Name>";
    private static String rsName = "<Ruleset Name>";

    public EngineInvoker() {
    }

    public static RuleEngine getRuleEngine() throws Exception {
        InitialContext context = new InitialContext();
        Object obj = context.lookup(jndiName);

        RuleEngineHome home = (RuleEngineHome) PortableRemoteObject.narrow(obj,
            RuleEngineHome.class);
        return (RuleEngine) home.create();
    }

    public static List invokeRuleset(<arguments to be passed into the ruleset for invocation>)
    {
        List input = createInputList(<arguments>);
        List output = new ArrayList();
        RuleEngine ruleEngine;

        if (projectName == null || rsName == null || input == null) {
            output.add("Project Name or Ruleset Name or Payload should not be NULL");
        }

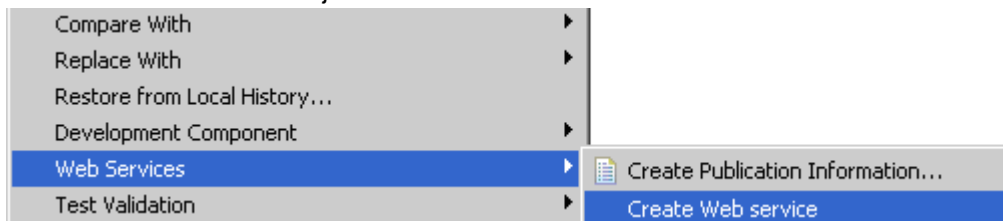
        try
        {
            ruleEngine = getRuleEngine();
            output = ruleEngine.invokeRuleset(projectName, rsName, input);
        }
        catch (Exception e)
        {
            output.add(e.getMessage());
        }
        return output;
    }

    private static List createInputList(<arguments to be passed to the ruleset for invocation>)
    {
        List input = new ArrayList();
        // Create the business object of the required type by using the arguments passed.
        Input.add(<business object>);
        return input;
    }
}

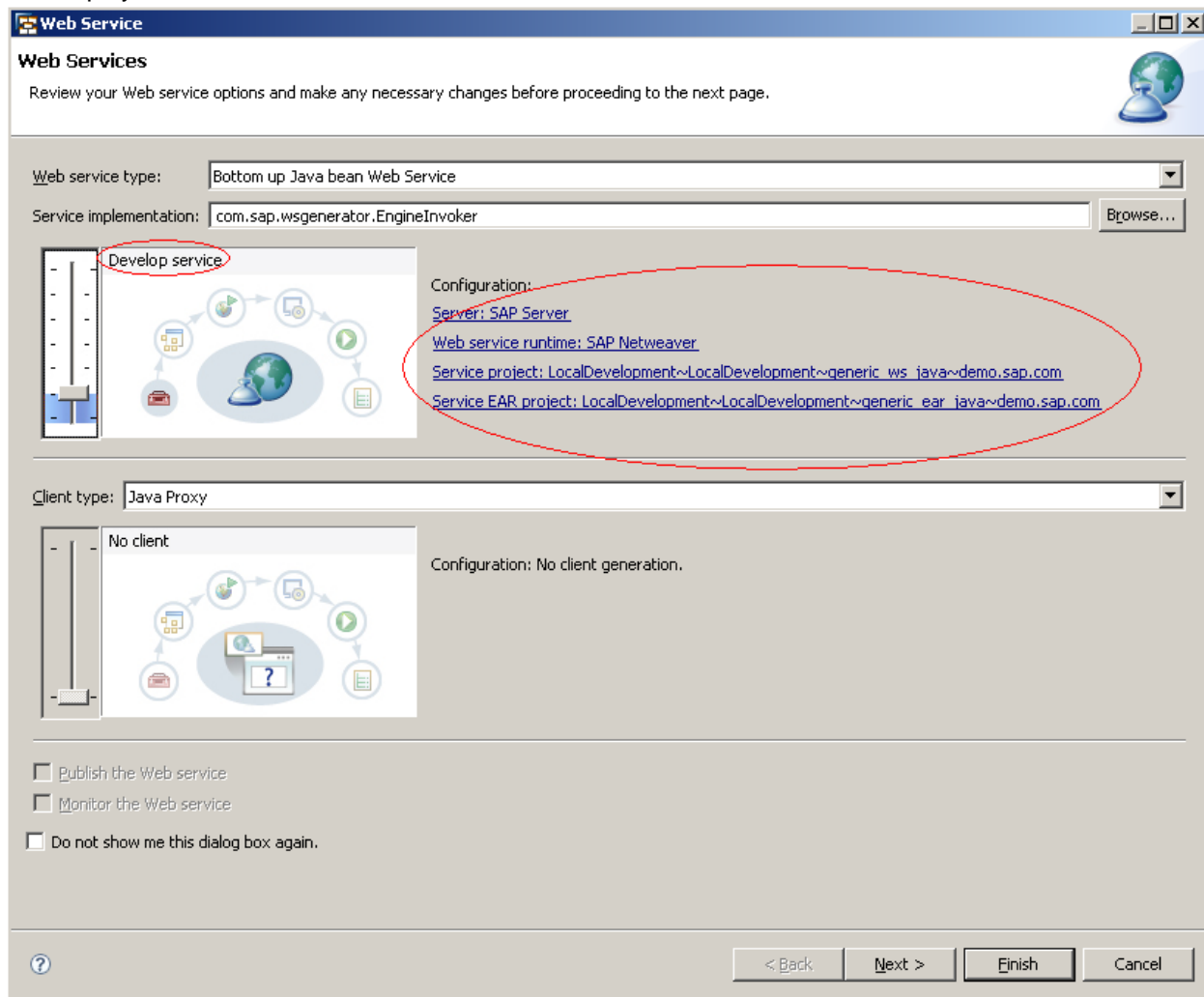
```

Creating Web Service

1. Now that the java class has been created, we need to generate a web service. To do this we make use of the web service generator tool in NWDS.
2. In the context menu of the java class created select *Web Services > Create Web Service*



3. In the web services window which appears, make the required configuration changes. The window displayed will be –

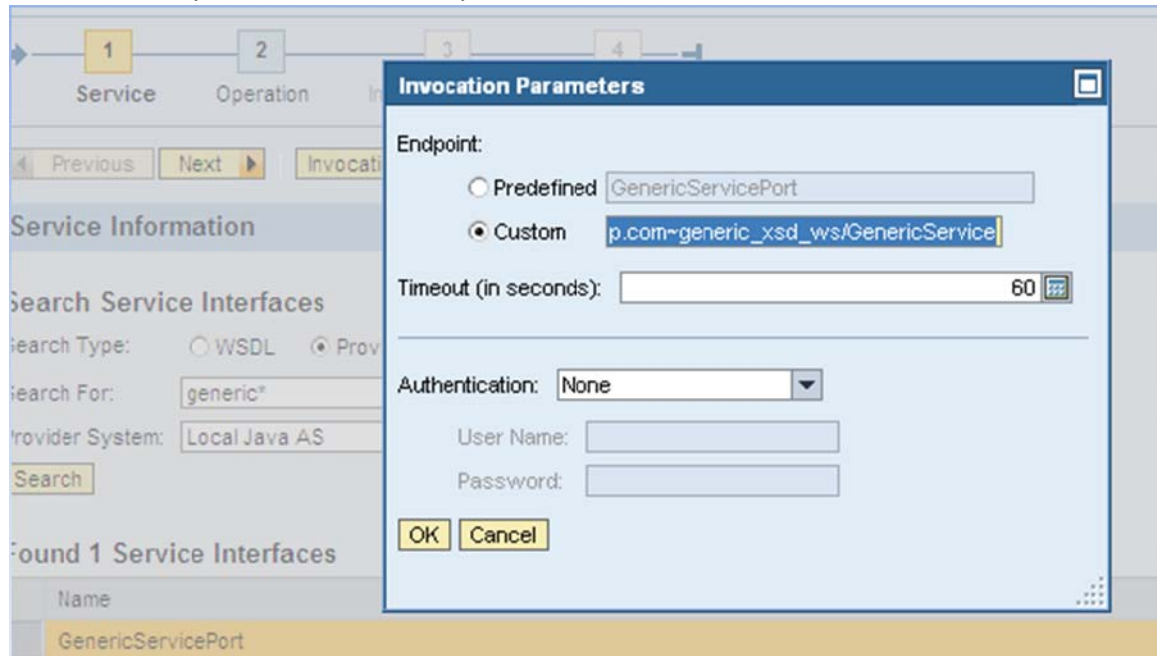


4. After the changes to the configuration has been made as required, select *Finish*. This will generate a web service
5. The web service generated needs to be build and deployed on to the Application Server. To do this –
 - a. First, we need to Build the web module DC
 - b. Then Build and Deploy the enterprise application DC.

Getting the WSDL from the WS Navigator

To get the wsdl of the webservice deployed into the Application Server–

- a. Access the **wsnavigator** of the portal – **http://<Application Server URL>: <port number>/wsnavigator**
- b. Select the **Provider System** radio button.
- c. Search for the service port name set in the java class.
- d. Select the Invocation Parameters of the service interface. Copy the url associated with the custom endpoint of the invocation parameters.



- e. Choose *Cancel*.
- f. Open the browser and use the link followed by *?wsdl* to access the WSDL file.

Example

An example for doing the same has been provided in SDN. Download the zip file from this location – <https://www.sdn.sap.com/irj/sdn/go/portal/prtroot/docs/library/uuid/2016a80c-a972-2b10-24bf-fcc87c9e653d>
This zip contains a rules project and the DCs created to generate the web service – a web module DC and an enterprise application DC.

Generating Web Service DCs

1. Extract the DCs – **loan_emi** (rules DC); **generic_ws_java** (a WebModule DC) and **generic_ear_java** (an Enterprise Application DC) from the zip file.
2. Copy these DCs into the Workspace's DC folder i.e into - `<Workspace>\jdi\LocalDevelopment\DCs\demo.sap.com`
3. Open your Workspace in NWDS.
4. Open *Development Infrastructure Perspective* in the Development Studio by *selecting Window > Open Perspective > Other > Development Infrastructure*.
5. In the *Component Browser View* you should be able to see the three DCs copied into the Workspace's DC previously.

Note: Refresh the folder if you cannot see the DCs.

6. Select the DCs. From the context menu of the DC *Sync/Create Project > Create Project*
7. In the window opened to *Sync Sources and Used DCs* click on *OK*.
8. In *Confirm Perspective Switch* dialog box select *Yes*

Customizing the DCs

Changes are to be made in the files provided to create a web service. Below is the list of modifications to be made -

1. Open the Web Module DC - **generic_ws_java**.
2. A class - **EngineInvoker.java** – has been created in this DC.
 - a. The web service provider information has been defined in the class. If required, changes can be made to the following parameters – *name, serviceName, portName and targetNamespace*

```
@WebService(name="RulesetInvoker", portName="RulesetInvokerPort", serviceName="RulesetInvokerService", targetNamespace="http://sap.com/ruleset/")
public class EngineInvoker {
```

- b. Open this class and make changes to the *projectname* and the *rulesetname* defined in the project.

```
private static String jndiName = "com.sap.brms.RuleEngine";
private static String projectName = "<Project Name>";
private static String rsName = "<Ruleset Name>";
```

3. Build the Web Module DC.
4. Build and Deploy the Enterprise Application DC.
5. You can get the wsdl by following the steps mentioned in the section before (Getting the WSDL from the WS Navigator)

Copyright

© 2008 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, System i, System i5, System p, System p5, System x, System z, System z9, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, Informix, i5/OS, POWER, POWER5, POWER5+, OpenPower and PowerPC are trademarks or registered trademarks of IBM Corporation.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

These materials are provided "as is" without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall not be liable for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials.

SAP does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within these materials. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third party web pages nor provide any warranty whatsoever relating to third party web pages.

Any software coding and/or code lines/strings ("Code") included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, except if such damages were caused by SAP intentionally or grossly negligent.