How To Guide: Consuming MII Web Service in Web Dynpro Java Using Adaptive Web Service Model

Applies to:

MII 12.0 (Build 110), NetWeaver CE 7.1

Summary

MII (MANUFACTURING INTEGRATION AND INTELLIGENCE) is part of SAP AG product in the NetWeaver suite. The technology has been applied to a wide range of industries and applications. It also supports SAP's adaptive manufacturing strategy. MII is typically used to link a shop floor system with an ERP and provide portals information from plant floor applications and systems. This exercise will give an idea how to consume MII Web Service in Java Web Dynpro, NW CE 7.1 using adaptive web service model.

Author: Pradip Ray
Company: SAP Labs India
Created on: 17 September 2008

Author Bio

Pradip Ray has been working with SAP Labs since 2006 and is part of the Manufacturing development team in Bangalore with main focus on manufacturing integration and collaboration projects. Before he moved to SAP, he worked for a steel manufacturing company and was involved in Level-1 and Level-2 automation projects. He has completed his Instrumentation Engineering from National Institute of Technology, Trichy.
# Table of Contents

- Introduction ................................................................................................................................................... 3
- Prerequisite .................................................................................................................................................... 3
- Create MII transaction and MII Web Service to get the MES Batch Data....................................................... 3
  - Create a BLS transaction with the SQL Query ........................................................................................... 6
  - Creation of Web service out of the transaction ......................................................................................... 11
  - Create An Adaptive Web Service Model: ................................................................................................. 11
  - Bind the used model with the component: ............................................................................................... 17
- Create Destinations in NWA (NetWeaver Administrator): ............................................................................. 25
- Create an Application: ................................................................................................................................... 28
- Final Display: ................................................................................................................................................ 29
- Copyright ....................................................................................................................................................... 30
Introduction:

Web Dynpro Java is a well accepted UI technology among SAP products. MII has a very good capability to communicate with the shop floor devices, software and Manufacturing Execution System. In this exercise we are going to use MII as a data source and Web Dynpro Java to display the information.

To know more about MII as a product Click on this link bellow:


In manufacturing area MES (Manufacturing Execution System) is closely integrated with the shop floor devices. It is assumed for this scenario that MES keeps the batch information for the products. We are going to display the MES Batch detail on Web Dynpro Java UI.

Prerequisite:

Knowledge of MII 12.0 ,Java Web Dynpro, NetWeaver CE 7.1

Web Dynpro Java Project Should be ready (As we are going to work with Model and small part in UI)

To create Web Dynpro Project follow the document:

https://www.sdn.sap.com/irj/sdn/articles-topic?rid=/webcontent/uuid/7082f9fc-070d-2a10-88a2-a82b12cea93c

Architecture:

As shown above, Web Dynpro Java application invokes the Web Service of MII transaction. MII transaction fetches data from MES DB. MII transaction uses different types of connector to communicate the Historian or MES Database. Here we used IDBC connector. All MII BLS transactions can be exposed as Web Services.

Create MII transaction and MII Web Service to get the MES Batch Data

To fetch the MES Batch data we are going to use SQL Queries from the MII. To do the same we need IDBC connection.

http://help.sap.com/saphelp_xmii120/helpdata/en/45/58d3e94d4009cae1000000a114a6b/content.htm link guides, how to make the IDBC connection from MII.

Now Go to SAP xMII work bench and follow the steps:
Step 1: Right click on the project and Create a SQL Query

Step 2: Make the Query as fixed Query
Step 3: Click on fixed Query Detail at left hand bottom panel

Step 4: Create a fix query at the left hand side

```sql
SELECT * FROM cake WHERE Ingredient LIKE 'flour' OR Ingredient LIKE '%sugar'
```
Step 5: Save the Query.

Create a BLS transaction with the SQL Query

The query template is ready now we have to create a Transaction with the Query Template

Step 1: Create Transaction.

Step 2: Create the logic as follows:
Step 3: Configure Object BatchAttrQuery (Refer the fig. above)

Step 4: Configure Object Repeater_for_batch_data:
Step 5: Configure Object XML_Structure.

![Image of Document Configuration dialog box with properties for 'AttrColor', 'MettingPoint', 'OrderID', 'ChipsPerGram', 'Material', 'Batch'.]

Step 6: Configure Link of Query_To_Document.

![Image of Link Editor dialog box with links to 'Query_To_Document'.]
Step 7: Create a transaction property as XML type and output parameter.

Step 7: Assign the output XML from transaction Context menu:
Step 8: Assign it to BatchAttrQuery.Result

Step 9: Configure the Link of assignment block:

Step 10: Save the transaction
Creation of Web service out of the transaction

http://<Host>:<port>/XMII/WSDLGen/<PathToTransaction> This URL will give the WSDL file for the transaction. Now we shall consume the same WSDL in Web Dynpro Java using adaptive web service model.

Consuming MII Web Services In Web Dynpro Model As Adaptive Web Service Model:

We are going to work on a already existing Web Dynpro Java project. If it is not ready then Create a Web Dynpro project it will look like this:

Create An Adaptive Web Service Model:

Step 1: Right click on the model and select create Model.

Step 2: Create model wizard, will navigate to the Model type selection screen. Select Adaptive Web Service Model and press “Next”.

Please select a Model Type

This wizard guides you to create an Adaptive Web Service Model.

- Adaptive RFC Model
- Adaptive Web Service Model
- Configuration Model
- Enterprise Services Model
- JMX Model
- Java Bean Model
- Web Service Model - DEPRECATED
Step 3: Give a model name and Model Package. Select “Next”

![New Model]

**Import Adaptive Web Service Model**

Create a Web Dynpro Model for a Web Service. You can select from the locally published Web Services or specify a WSDL file.

<table>
<thead>
<tr>
<th>Model Name</th>
<th>MiiBatchModel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model Package</td>
<td>com.sap.miibatch.model</td>
</tr>
</tbody>
</table>

Step 4: In logical destination Use the second option and Set the Default Metadata Destination and Default Execution Destination. We have to configure the same in NetWeaver Administrator. Select “Next”

![New Model]

**Logical Destinations**

Define logical destinations for metadata retrieval and execution

- No logical destinations - use WSDL URL for metadata retrieval and webservice execution
- Use destinations for metadata and execution - requires provider to support WSIL, e.g. supported by SAP Systems

<table>
<thead>
<tr>
<th>Default Metadata Destination</th>
<th>MII_Dest</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Execution Destination</td>
<td>MII_Exec</td>
</tr>
</tbody>
</table>

You can change the destination names after import via changing the corresponding model setting values. The destinations have to be maintained in the J2EE Visual Admin.
Step 5: Give the WSDL file or URL to create the Adaptive Web Service model. Give MII user name and password. Select “Next”.

![Image of WSDL file selection](image1)

Choose a WSDL file from the file system or via URL

<table>
<thead>
<tr>
<th>WSDL</th>
<th>http://&lt;HostName&gt;:&lt;Port&gt;/MII/WSDLGen/&lt;PathToTransaction&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td>User Name</td>
</tr>
<tr>
<td>Password</td>
<td>**********</td>
</tr>
</tbody>
</table>

Provide User Name and Password for WSDL access if needed.

Step 6: Select next with default Name space and Entity name.

![Image of Rename Name Spaces and Model Classes](image2)

Rename Name Spaces and Model Classes.

This wizard page lets you rename namespaces and entity names.

**Namespaces**

<table>
<thead>
<tr>
<th>NameSpace</th>
<th>Editable Alias</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://sap.com/tc/webdynpro/model/webservice">http://sap.com/tc/webdynpro/model/webservice</a></td>
<td></td>
</tr>
<tr>
<td><a href="http://www.sap.com/">http://www.sap.com/</a>&lt;MII&gt;</td>
<td></td>
</tr>
</tbody>
</table>

**Model Classes**

<table>
<thead>
<tr>
<th>Entity Name</th>
<th>Editable Model Class Name</th>
<th>Final Model Class Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputParams</td>
<td>InputParams</td>
<td>InputParams</td>
</tr>
<tr>
<td>Row</td>
<td>Row</td>
<td>Row</td>
</tr>
<tr>
<td>Rowset</td>
<td>Rowset</td>
<td>Rowset</td>
</tr>
<tr>
<td>ExecuteRequest</td>
<td>ExecuteRequest</td>
<td>ExecuteRequest</td>
</tr>
<tr>
<td>ExecuteResponse</td>
<td>ExecuteResponse</td>
<td>ExecuteResponse</td>
</tr>
</tbody>
</table>

Replace All | Restore Default
Step 7: Select “Next”

Now we have created a Adaptive Web Service model from the MII Web Service.

Step 8: Select “Finish” Button.

Now we have created a Adaptive Web Service model from the MII Web Service.
Create a Component:
Right click on the component and create new component.
Step 1: Give a component name and package. Select the checkbox Default window and views.
Select checkbox Used Models.

Step 2: We can select the default names for the windows and the views.
Step 3: Select the ADD button

Step 4: Select the Model and press “OK” Button
Bind used model with the component:

To bind the model with component follow the steps.

Step 1: Right click on the component controller and apply template:

[Diagram showing steps to bind model with component]

Step 2: Select Service controller

[Window showing select template dialog]

Step 3: Select the executable model class
Step 4: Bind the context elements

Select the context elements you want to bind. Rename them by clicking the corresponding item in the list on the right.
As we are using adaptive web service model, we do not need to pass user name and password in the WSDL input. For this scenario we are taking only response.

Step 5: Generate the method for model Execution. Select “Finish” Button

Step 6: Check the coding part:

Open the <component controller name>.java (From Navigator tab The file will be under the project ➔ gen_wdp ➔ packages)
Check the method `wdDoInit()` the coding has to look like it is given bellow:

```java
public void wdDoInit()
{
    //@@begin wdDoInit()
    //$$begin Service Controller(-478203907)
    MiiBatchModel model = new MiiBatchModel();
    wdContext.nodeRequest_Xacute().bind(new Request_Xacute(model));
    //$$end
    //@@end
}
```

Check the method `executeXacute()` the coding has to look like given bellow:

```java
public void executeXacute()
{
    //@@begin executeXacute()
    //$$begin Service Controller(2142827532)
    IWDMessageManager manager = wdComponentAPI.getMessageManager();
    try
    {
        wdContext.currentRequest_XacuteElement().modelObject().execute();
        wdContext.nodeResponse().invalidate();
    }
    catch(Exception e)
    {
        manager.reportException(e.getMessage());
    }
    //$$end
    //@@end
}
```

We are going to display the MES detail on the UI. So we are going to call the model execution method at `wdDoInit()` of component controller. We can also call this method any where from View using `wdThis.wdGet<Component Controller Name>Controller().executeXacute()`.

For our scenario code will go into component controller `wdDoInit()`

```java
public void wdDoInit()
{
    //@@begin wdDoInit()
    //$$begin Service Controller(-478203907)
    MiiBatchModel model = new MiiBatchModel();
    wdContext.nodeRequest_Xacute().bind(new Request_Xacute(model));
    wdThis.executeXacute();
    //$$end
    //@@end
}
```
Context Mapping of Component controller and View:

Step 1: Create a data link between component controller and View

Step 2: Map component controller context elements in View context
Step 3: Select Finish button

Step 4: To Create the UI we are going to apply template. Right click on the View and Apply template.
Step 5: Select Table from Template

Step 6: Select the attributes for the table.
Step 7: Arrange the table elements.

A default column header text is set in case there is no other type information available. In some special models this could hide potential column header metadata at runtime.

- Set default header texts for all columns

Step 8. Select Finish button.
Create Destinations in NWA (NetWeaver Administrator):

During the creation of Adaptive Web Service model using the MII web service we have given Default Metadata Destination as MII_Dest and Default execution Destination as MII_Exec. Now we have to create both the destinations in NWA. The URL for the NWA is:

http://<Host>:

Step 1: On the NWA page go to soa manager. (Shown Below)

Step 2: At technical Configuration tab, click on Destination Template Management.

Step 3: Configure a new Destination

Configure Destinations

New Delete

Step 4: Give the detail as shown below. In URL part we give “XMII/WSDLGen”. This part is responsible for metadata. Select “Next”.

New Destination

1 2

General Security

Enter a name for the logical port

Destination Type: WSDL

Destination Name: MII_Dest

URL: http://<Host>:<Port> XMII/WSDLGen/<TransactionPath>

Socket Timeout (in milliseconds): 60,000

System: Java

Hostname: <HostPort>
Step 5: Select the Authentication as “HTTP Authentication”. Select the radio button “User ID/Password (Basic). Select the Button “Details On”, give the user ID and Password. Keep rest of the thing as default and Press “Finish” Button.

The Configured destination looks like this.
Step 6: Create another Destination as shown below. Here in URL we give “XMII/SOAPRunner” this executes the web service of MII. Select “Next”

![New Destination form](image)

Step 7: Follow step 5 and finish.
Create an Application:

Now we have to create an application. Right click on the application and Create the application.

Step 1: Give the application name and package.

Step 2: Use existing components. Select “Next”.

Step 3: Select the components. Press “Finish”.

Step 4: Right click on the project → Rebuild Project

Step 5: Right click on the application → Select Deploy new Archive and run.

**Final Display:**

<table>
<thead>
<tr>
<th>ChipsPerGram</th>
<th>OrderNo</th>
<th>Color</th>
<th>MeltingPt</th>
<th>LotID</th>
<th>BatchID</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>ORDER00001</td>
<td>RED</td>
<td>285</td>
<td>BH_SEMI_1</td>
<td>BH_1</td>
</tr>
<tr>
<td>22</td>
<td>ORDER00002</td>
<td>BLUE</td>
<td>280</td>
<td>BH_SEMI_1</td>
<td>OR_1</td>
</tr>
<tr>
<td>41</td>
<td>ORDER00003</td>
<td>YELLOW</td>
<td>315</td>
<td>BATCH_FIN_01</td>
<td>0000000077</td>
</tr>
<tr>
<td>49</td>
<td>ORDER00003</td>
<td>GREEN</td>
<td>346</td>
<td>BATCH_FIN_01</td>
<td>0000000078</td>
</tr>
<tr>
<td>65</td>
<td>ORDER00004</td>
<td>BLUE</td>
<td>310</td>
<td>BATCH_FIN_01</td>
<td>0000000130</td>
</tr>
</tbody>
</table>

Note: SAP does not provide support for this application; all risk is assumed by the customer. SAP is not liable for consequences of or damages resulting from the customer’s use of the application.
How to Guide on consuming MII Web Services using Adaptive Web Service Model in Web Dynpro Java

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.