Using SAP MII as Data Source and Environment for Business Objects Xcelsius

Applies to:

SAP MII 12.0/SAP BusinessObjects Xcelsius 2008. For more information, visit the Manufacturing homepage.

Summary

This article provides a guide on how to connect SAP MII 12.0 to SAP Business Objects Xcelsius in order to visualize manufacturing execution data in Xcelsius dashboards which are embed into the SAP MII portal

Author: AN Navaneeth and Stephan Boecker

Company: SAP

Created on: 05 September 2008

Author Bio

AN Navaneeth and Stephan Boecker work in Solution Management Manufacturing at SAP with a focus on SAP MII.

Table of Contents

Introduction

The acquisition of Business Objects into the SAP product family created the need to integrate product components to allow customers to implement new scenarios. In the manufacturing area a close integration between SAP MII and the SAP Business Objects product suite is part of the actual SAP MII roadmap. One aspect of this integration is the creation of a specific SAP MII connector for the SAP Business Objects suite of products which will allow Business Object users to access SAP MII and its composition environment as a new data source. At the same time SAP MII will allow plant personnel to take advantage of Business Objects elements inside the SAP MII environment. However, most of the integration pieces described in the roadmap are meant for a long term strategy and do not serve the immediate need to integrate these SAP components. Especially in most scenarios involving dashboards, like the Business Objects Xcelsius product, customers need how-to guides on how to drive their own development. Combining Xcelsius's appealing user interface environment with SAP MII's data connectors is an interesting combination of a user interface modeling and data modeling tool combined to achieve maximum flexibility for customer dashboard applications. The goal of this article is to show customer developers how to use SAP MII as data source for an Xcelsius dashboard. The article explains current workarounds in order to achieve this based on the existing Xcelsius and SAP MII versions.

Connectivity

Connection Type and Structure

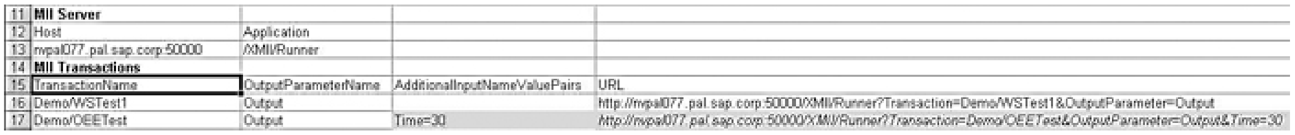
The easiest and most flexible way to establish connectivity from Xcelsius to a SAP MII server is to use the XML data connector pointing to a transaction you want to be executed on the SAP MII server. The output of the transaction will be an XML data stream in a format usable by Xcelsius. The typical URL to launch a transaction in SAP MII is assembled from different elements and looks like the following:

http://<server>:<port>/XMII/Runner?Transaction=<transaction>&OutputParameter=<output>&<InputParameterName1>=<InputParameterValue1>&>&<InputParameterName2>=<InputParameterValue2>... where

- <server> is the name of the SAP MII host.
- <port> is the port of the SAP MII application.
- <transaction> is the name of the transaction as part of the SAP MII workbench you want to be executed. You have to use the full path name to the transaction starting from the project folder. A transaction located in "MyProject" with a name of "TestTransaction" would be referred to as "MyProject/TestTransaction".
- <output> is the name of the transaction output parameter which contains the return XML to be used in Xcelsius. This parameter has to be of type XML
- <InputParameterName1>, <InputParameterName2> are the names of input parameters to the transaction you want to fill before the transaction is executed. These parameters have to be of a primitive type and are optional.
- <InputParameterValue1>, <InputParameterValue2> are the values associated with the input parameters mentioned above. The values are optional.

Setting up the Connection

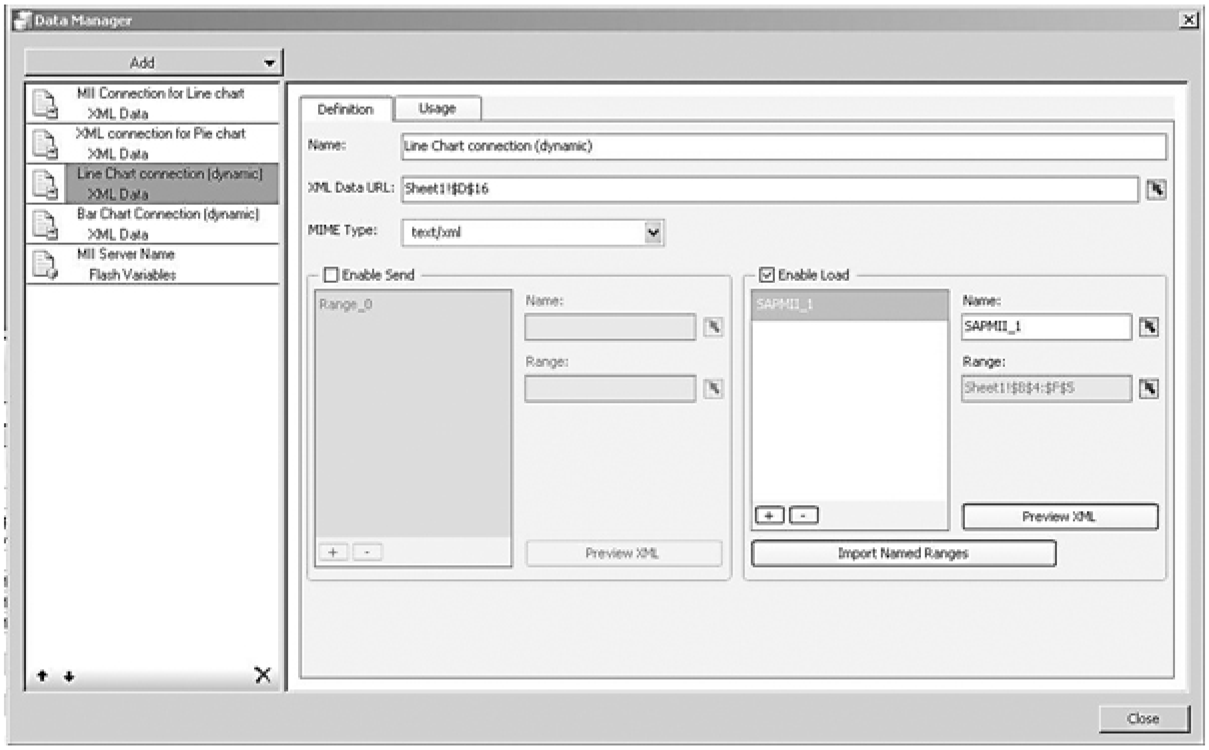
In Xcelsius you get the most flexibility by using the Excel sheet to assemble the URL to the SAP MII server with the structure explained above, especially if you intend to use several calls to SAP MII in your dashboard. In the screenshot below you can see an example how those URLs are assembled:



11	**MII Server**			
12	Host	Application		
13	nvpal077.pal.sap.corp:50000	/XMII/Runner		
14	**MII Transactions**			
15	TransactionName	OutputParameterName	AdditionalInputNameValuePairs	URL
16	Demo/WSTest1	Output		http://nvpal077.pal.sap.corp:50000/XMII/Runner?Transaction=Demo/WSTest1&OutputParameter=Output
17	Demo/OEETest	Output	Time=30	http://nvpal077.pal.sap.corp:50000/XMII/Runner?Transaction=Demo/OEETest&OutputParameter=Output&Time=30

Note: The second URL on line 17 shows an example how an additional input parameter is used. The value is taken from an Excel cell filled via user interaction. The URL is a simple concatenation performed by the Excel worksheet based on the server and transaction information

The Data connection in Xcelsius is referring to the Cell containing the assembled URL:



Name: Line Chart connection (dynamic)

XML Data URL: Sheet1!$D$16

MIME Type: text/xml

☐ Enable Send — Name: Range:

☑ Enable Load — SAPMII_1 — Name: SAPMII_1 — Range: Sheet1!$B$4:$F$5

Building the Application

Build a dashboard in Xcelsius

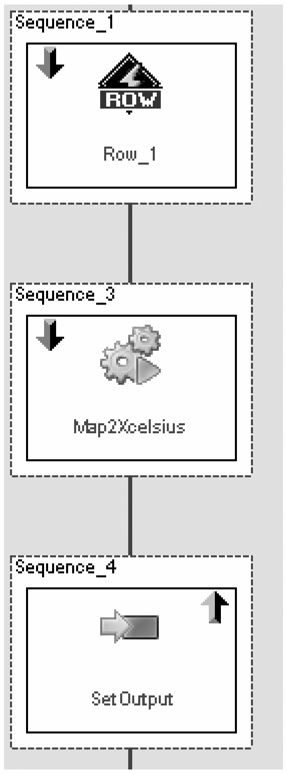
Once you have established the connectivity to SAP MII, you can create the visualizations needed. Controls should refer to the range in the spreadsheet which will be filled by SAP MII at runtime.

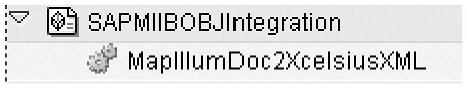
Matching the XML Structures

Xcelsius expects a specific XML format for a target range which can be viewed by pressing the 'Preview XML'. The XML format contains a simple set of row elements according to the target range chosen and <column> elements as children of these rows. SAP MII uses a different slightly more complex XML format for internal use. In order to facilitate the transformation from this 'IllumXML' used by SAP MII to the format expected for a range of cells in Xcelsius a mapping transaction has been provided which performs the following operations:

- Each <Rowset> in a SAP MII output will be mapped to a new variable in Xcelsius based on the positioning the document, starting with 'SAPMII_1'. The second <Rowset> therefore would be a new variable named 'SAPMII_2' etc.
- Each <Row> inside a <Rowset> will map to a <row> in Xcelsius.
- Each element which is a child of a <Row> will map to a <column> element in Xcelsius. The value of the element will be transferred to the <column> text.

This generic mapping will allow you to use 'IllumXML' inside your SAP MII transaction. Before you assign the final result to your transaction output you can use the 'Call transaction' action block to invoke the mapping as shown in the following screenshot:

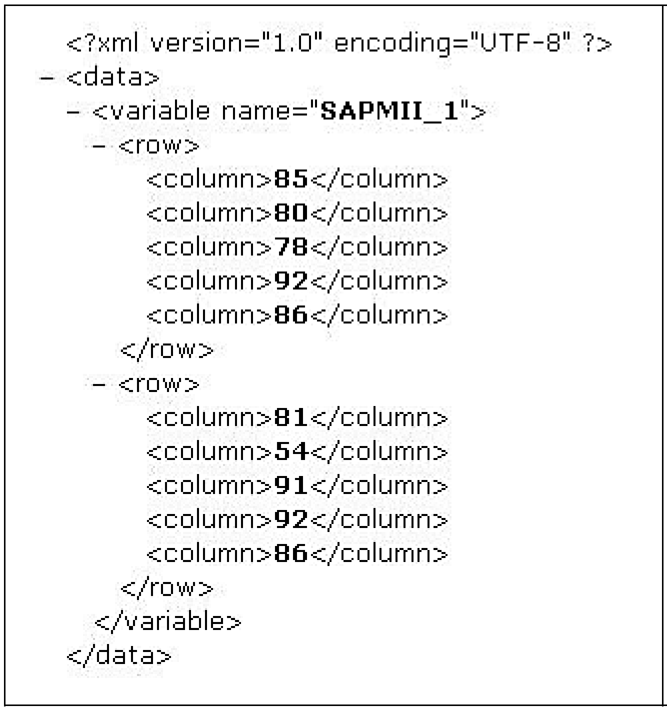




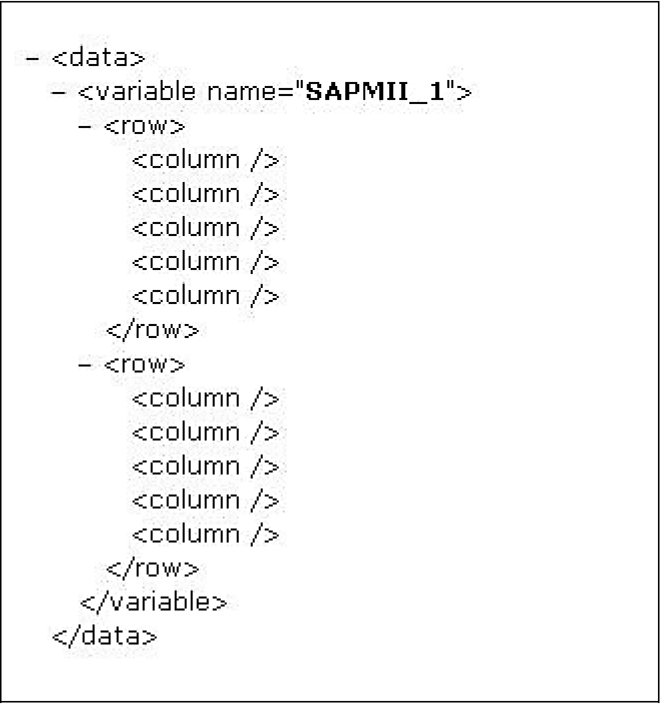
The mapping transaction is provided as part of the attached SAP MII project 'SAPMIIBOBJIntegration' which you can import into your SAP MII server environment.

Testing the Transaction Output

The easiest way to test the SAP MI transaction is to use the XML data URL value in your browser. If your transaction works correctly you should receive a logon screen to SAP MII. After you authenticate you should receive an XML response document (shown in the left side of the screen shot below) which structurally matches the result you get from the 'Preview XML' button in the Data Manager dialog (shown on the right in the screen shot below).



```
  <?xml version="1.0" encoding="UTF-8" ?>
- <data>
  - <variable name="SAPMII_1">
    - <row>
        <column>85</column>
        <column>80</column>
        <column>78</column>
        <column>92</column>
        <column>86</column>
      </row>
    - <row>
        <column>81</column>
        <column>54</column>
        <column>91</column>
        <column>92</column>
        <column>86</column>
      </row>
    </variable>
  </data>
```



```
- <data>
  - <variable name="SAPMII_1">
    - <row>
        <column />
        <column />
        <column />
        <column />
        <column />
      </row>
    - <row>
        <column />
        <column />
        <column />
        <column />
        <column />
      </row>
    </variable>
  </data>
```

Previewing the Application

If you try to preview the application inside Xcelsius you will not receive the XML output you expected, since the response from the SAP MII server to your preview request in Xcelsius will be the SAP MII logon page. This problem only exists inside Xcelsius, not after importing the resulting .swf file into the SAP MII server as explained in the next section. So how can you bypass authentication inside Xcelsius? The specific SAP MII application triggering transactions (the 'Runner') you are calling allows you to attach a user name and password as part of the URL for test purposes. The parameters are called 'XacuteLoginName' and 'XacuteLoginPassword'. If you enhance the Excel sheet to include these parameters as part of your URL, you should be able to preview your application in Xcelsius and receive the XML data expected.

Note: Make sure you do not use this feature for productive scenarios. Always delete the part of the URL containing authentication information before exporting the .swf file to the SAP MII server. Otherwise all users launching the .swf file will use your credentials to authenticate instead of the SAP logon ticket, and will transmit them as part of the URL in clear text.

Direct Connection to a SAP MII Query

In case you do not need the flexibility of a SAP MII transaction as discussed previously you can directly use the result of a query. The URL used would refer to the 'Illuminator' service and look like:

http://<server>:<port>/XMII/Illuminator?QueryTemplate=<QueryTemplate>&Content-Type=text/xml where

- <QueryTemplate> is the name of the query template you want to be executed.

In order to achieve the correct output you would need to define the XML mapping as an inline transformation in your query definition or add to the URL dynamically. The parameters to preview the result in the Xcelsius application are names 'IllumLoginName' and 'IllumLoginPassword'.

Direct Connection to a SAP MII Web Page

Using the URL Button in the Web Connectivity block in Xcelsius you can directly navigate to a web page hosted on SAP MII. In a typical scenario you would use this forward navigation to see plant level details for a deeper analysis of data shown in your Xcelsius dashboard. The path to the specific web page can be found by using the 'Test' button in the SAP MII workbench for that page and assembling the URL similar to the procedure described before. Previewing the application should provide you with a logon page followed by the requested page. At runtime, users should be able to directly navigate to the requested page.

Logistics

Exporting the Application

Export your Xcelsius application by choosing File->Export->HTML from the menu. This will provide you with an .html and a .swf file on you local machine.

Importing the Application into SAP MII

In the SAP MI workbench, navigate on the 'Web' tab to the project you want to use and choose 'Import' from the right mouse context menu. In the 'Open' dialog window, choose the .html and .swf file created in the previous step and import them by pressing the 'Open' button. As a result both pages should now be part of your SAP MII project. You can check and edit the .html file to apply any changes you want as long as you don't change the block embedding the .swf file. If you want to use SAP MII session variables you can also change the ending of the .html file to .irpt and provide more advanced changes such as combining Xcelsius dashboards with dynamic native SAP MII UI elements in a single page. If you launch the web page from the workbench via the 'Test' button, you should see the web page including your dashboard.

Embedding the Application into the SAP MII portal

Since the Xcelsius application is part of a .swf file included in the web page you import, the usage in the SAP MI portal is exactly the same as any other web page. Use the 'Navigation Services' to put the web page into the role based SAP MII portal tree. Once end users access log on to the SAP MI portal their browser session will receive a SAP logon ticket from SAP MII in form of a cookie which will be used for all subsequent requests from the .swf file to SAP MII server for data acquisition.

Exporting and Embedding into other Environments

You can embed the .html and .swf file as well into other SAP Netweaver based web environments, such as SAP Enterprise portal or a J2ee Web Dynpro application. Make sure the logon ticket issued by those environments is accepted by the J2ee engine hosting SAP MII as described in the SAP Netweaver documentation. Embedding into other environments will leave you with the problem of authentication from the .swf file to the SAP MII server in the same way as described above for the Xcelsius application itself.

Note: Be aware that browser restrictions apply for the forwarding of cookies across domains. In addition the flash player which is part of your end user client environment will not allow a .swf file to receive data from a SAP MII server in another domain than the host where the .swf file was loaded from. Please refer to the SAP Netweaver and Adobe Flex documentation on further details

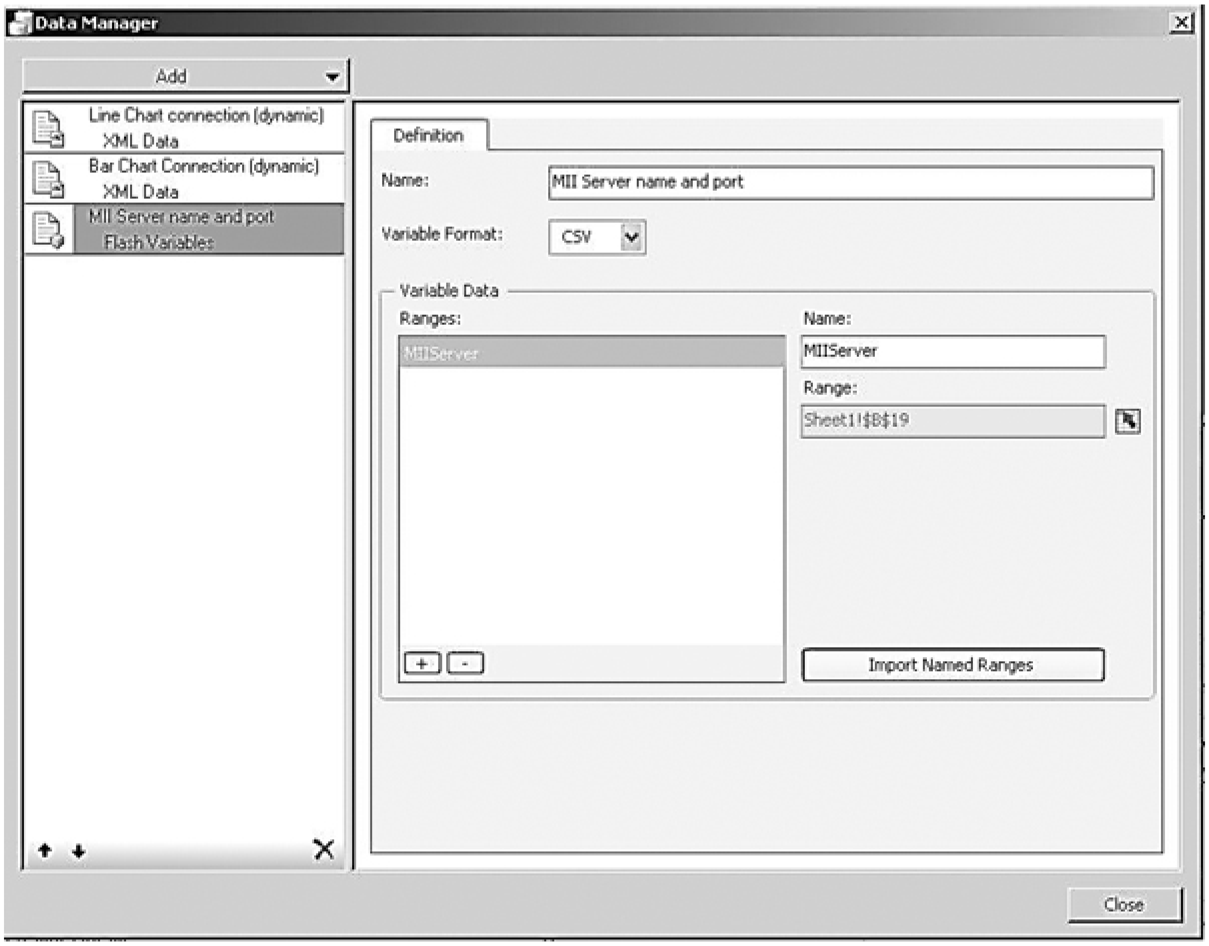
Process flow

For a .swf file embedded into a SAP MII web page the process flow is as follows:

1. The user logs on to SAP MII and receives his/her role based portal view and a login ticket.
2. Once the user clicks on the web page, the web page will load. At the same time the browser of the user will load the .swf file embedded as part of the web page.
3. Whenever the event is fired to collect data from SAP MII, the Xcelsius application will use the URL's you created for data access to SAP MII. Since the user is already logged on to SAP MII his logon ticket will be used to retrieve data.
4. SAP MII will return the XML data needed.
5. The .swf file will display the data

Using flash Variables to Increase Flexibility

So far our scenario involved receiving data from a dedicated SAP MII server via a URL. The .swf file exported from Xcelsius is imported into a possible different SAP MII server. Ideally you could use the server hosting the .swf file to be the same as the one used for data retrieval. This is especially beneficial for scenarios where you use a development, test, and productive environment and don't want to have to access Xcelsius every time to change the underlying server used. In this scenario, flash variables can give you the flexibility needed. Start by defining flash variables inside Xcelsius which bind to an Excel worksheet cell as shown in the following screenshot.



Name: MII Server name and port

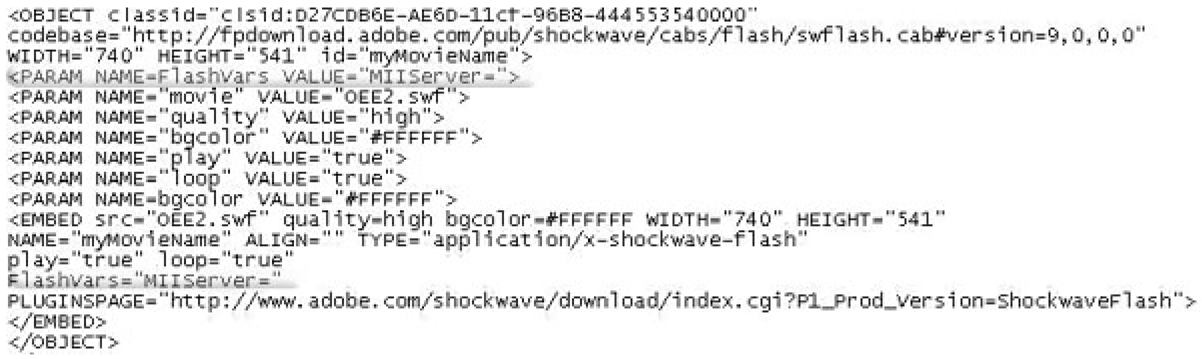
Variable Format: CSV

Ranges: MIIServer

Name: MIIServer

Range: Sheet1!$B$19

Inside Xcelsius use the server name and port information contained in this cell to adjust your URL's pointing to the SAP MII server for data access. The real server name and port will later be passed into the .swf file as a flash variable. Before exporting make sure the cell value referenced is empty. After exporting your .html page refers to the flash variable defined before:



```
<OBJECT classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://fpdownload.adobe.com/pub/shockwave/cabs/flash/swflash.cab#version=9,0,0,0"
WIDTH="740" HEIGHT="541" id="myMovieName">
<PARAM NAME=FlashVars VALUE="MIIServer=">
<PARAM NAME="movie" VALUE="OEE2.swf">
<PARAM NAME="quality" VALUE="high">
<PARAM NAME="bgcolor" VALUE="#FFFFFF">
<PARAM NAME="play" VALUE="true">
<PARAM NAME="loop" VALUE="true">
<PARAM NAME=bgcolor VALUE="#FFFFFF">
<EMBED src="OEE2.swf" quality=high bgcolor=#FFFFFF WIDTH="740" HEIGHT="541"
NAME="myMovieName" ALIGN="" TYPE="application/x-shockwave-flash"
play="true" loop="true"
FlashVars="MIIServer="
PLUGINSPAGE="http://www.adobe.com/shockwave/download/index.cgi?P1_Prod_Version=ShockwaveFlash">
</EMBED>
</OBJECT>
```

If you assign a value to the 'MIIServer=' variable like 'MIIServer=myserver.mycompany.com:50000', the value will be passed into the .swf file as variable and form there used inside the Excel cell defined above. You can use web page scripting to dynamically read the server and port of the actual web page, use another applet on the page, or any other way to extract this information. If you are only interested in rendering the .html file and your dashboard, the easiest way we think is to save the .html page as .jsp page and extract server and port form the java request object similar to the following code

```
<%

String server = request.getServerName();

String port = String.valueOf(request.getServerPort());

String host= server + ":" + port;

%>
```

And insert the value in the call to embed the .swf file like

```
…

<PARAM NAME=FlashVars VALUE="MIIServer=<%=host%>">
….
```

Usage tips

Since SAP MII and SAP BusinessObjects Xcelsius are both more tool sets than ready to use software applications, it is pointless to limit the amount of scenarios you can cover by combining both tools. However, from our experience we would like to share the following findings:

- If you are a SAP MII experienced user, you surely will enjoy the modern user interface provided by Xcelsius. Another point worth noting is how easy it is to use the built-in Excel worksheet to provide logic for display/interaction compared to the scripting approach used in SAP MII.
- If you are an Xcelsius experienced user, you should try to leverage SAP MII as much as possible for data aggregation, combination, and manipulation, instead of trying to perform these logic steps inside Xcelsius. The reason is that SAP MII has been built to use and combine data sources ranging from fine granular tag data to ERP/Data warehouse data. Handling those data in a powerful server environment is almost always beneficial to trying to perform these steps in an end user client environment. In addition this will give you the opportunity to focus on data visualization logic in Xcelsius making your Xcelsius application easier to manage.

Copyright