Metadata Cache Invalidation for Adaptive RFC Models

Applies to:
Adaptive RFC model is available with Web Dynpro for Java. The article provides the metadata cache invalidation solution for NW 6.40, 7.0 and 7.1 releases.

Summary
The Adaptive RFC Model provides a Web Dynpro model implementation for backend R/3 System. Customers have regularly complained that the model is not adaptive enough to changes on the R/3 System unless the J2EE Engine is restarted. This article makes an attempt to solve this problem.

Author: Arun Bhat J
Company: SAP Labs India Private Ltd.
Created on: 30 April 2007

Author Bio
Arun Bhat works as a Development Lead at SAP Labs in the Web Dynpro UI Development area. He earlier worked in the Mobile Web Dynpro Offline Client area for two years since he joined SAP in Feb 2004. Currently he has been working on Web Dynpro models, Context and other related areas of the Web Dynpro framework.
1. Current Problems with the Adaptive RFC Model

The Adaptive RFC Model currently available with NW 6.40, 7.0 and 7.1 Releases may not always adapt to changes in the metadata, that is data types/metadata of the R/3 system from which the model pulls data. Maybe 2 simple scenarios should explain in greater detail the problem.

1. The tool tips on the Adaptive RFC are pulled from the R/3 Backend. Something like below. See “Value of Attribute” which is a tool top that shows up for the “TestAttr” input field.

Fig1. Tool Tips read from the backend through the Data Dictionary

If this were changed on the R/3 as shown in the Short Description as “Value of Test Attribute CHANGED!” (Fig2. Changing the description on the R/3) then this should also be visible in the Fig1 shown above, but it does not. The output continues to be the same.
Fig 2. Changing Data Element Descriptions on the R/3 Backend

2. Let's say there are some changes on the BAPI side itself. For example that the import/export parameters of the BAPI get changed.

For example the “TEST_ATTR” of the import parameter of the below BAPI Changed. Then the current solution doesn't really work.
Fig3. Renaming of the attribute of a BAPI

When the application is launched and run it rather doesn’t find the new “TEST_ATTR1” type and the program crashes like with the error below.

Fig3 Error after running a re-imported model currently without server restart
2. Reasoning why the metadata is not refreshed

The Web Dynpro Adaptive RFC framework sits on the top of several layers such as the Java Connector Objects (JCO), Data Dictionary (DDIC) and the MDI.

After the first access each of these layers caches the metadata within them and keeps returning the cached values thereafter. The caches are not refreshed even if something would have changed in the metadata on the R/3 backend. In addition the Web Dynpro ARFC model itself also caches the metadata. The new invalidation tool will try to refresh the caches across all the layers. The earlier solution to refresh the caches was to simply restart the engine.

3. Scenarios for Cache Invalidation

Customers have two scenarios across which they can use the Adaptive RFC Metadata Cache invalidation tool. They are

1. Development Scenarios – These scenarios involve not adapting the metadata to a running productive system. They are scenarios with respect to development where the Adaptive RFC model will have to be re-imported on the IDE and the application re-deployed on the engine. The redeployment is important because it also clears the Web Dynpro’s Adaptive RFC Cache.

   The scenarios are usually those that you would refresh the JCO Caches. The examples being
   a. Addition of import/export parameters/table parameters to the BAPI
   b. Deletion of import/export parameters/table parameters of the BAPI
   c. Renaming of existing parameters of a BAPI.
   d. Change of associated type of a parameter in the BAPI.

   All of these scenarios would need a “MODEL RE-IMPORT” on the IDE. The binding in the application will then need to be re-configured and adjusted with the new attributes in the model classes. The application then needs to be built and re-deployed.

   Before redeploying the application the ARFC Metadata Cache Invalidation application needs to be launched and run.

   Once there follow the following steps.
   a. The screen shows up something like this.
b. Once here click on the “Get JCO Destinations”. This will list all the JCO Destinations. There is a check box in the second column which indicates “Don’t Cache”. This means for that JCO destination which uses the BAPI, checking the box will not cache any metadata and always get it from the backend. Remember this doesn’t cache any metadata and always gets all metadata from the backend. This should be used only for the first access of the BAPI after the metadata on the R/3 has been changed (that is running of the actual application). After the “Don’t Cache” has been checked and the actual application is run we now access the changed ARFC metadata. After this you can uncheck the “Don’t Cache”, so that the Caching becomes active. This will help in improved Performance. For Production Scenarios NEVER CHECK THE “Don’t Cache!”

Please remember you always only check the Metadata JCO Destination only.
c. Next step is also to invalidate the related Data Dictionary Caches. Choose the metadata JCO Destination in question and click on "Get Dictionaries for Selected JCO Destination". This will list all the Data Dictionaries of that JCO Destination. Check the invalidate Cache for all dictionaries and click on the "Invalidate Selected Dictionary Cache" button. This will clear the Data Dictionary Cache. This just invalidates the cache and doesn't necessarily get the metadata every time from the backend. So this is not similar to Point b above.

There are 2 different ways of using the Dictionary invalidation. These will be discussed in the next topic.

---

**Fig 5. Non caching of the JCO Metadata Destinations**

There are 2 options to flush the dictionary Cache.

1. *Select dictionaries for the previously selected JCO Destinations*

   For performance reasons it is recommended to use the second option.

2. *Enter dictionary name manually into the name field for which to invalidate the cache.*

   The name can be derived from the model name for "'com.sap.test.MyModel'" it is "com.sap.test.types.MyModel".

---

**Dictionaries Cache Invalidation**

There are 2 options to flush the dictionary Cache in the tabs below.

**Option 1:** Select the Dictionaries for the currently selected JCO Destination and invalidate the dictionary cache.

This is the best option for performance. For performance reasons, use option 2 below.

**Option 2:** Enter dictionary name manually into the name field for which to invalidate the cache. The name can be derived from the model name for "'com.sap.test.MyModel'" it is "com.sap.test.types.MyModel".

---

**Get Dictionaries for Selected JCO Destination**

<table>
<thead>
<tr>
<th>Dictionary</th>
<th>Invalidate Cache</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.sap.test.types.Text</td>
<td></td>
</tr>
<tr>
<td>com.sap.mymodel.types.MyModel</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Invalidate Selected Dictionary Cache**
2. Production Scenarios – The production scenarios are those where any productive ARFC Web Dynpro system running in a customer environment can adapt to the metadata changes just by a browser refresh. There will be no need to “Restart the J2EE Engine”. Example Scenarios are those that involve refreshing the Data Dictionary Cache. Some of these are:

1. “Short Description” change of Data Elements.
2. Changes in Value Range of Domains on the backend etc.
3. Mostly things that have anything to do with Dictionary.

After running through the invalidation tool (as in Fig 6) the browser in which the application runs can simply be refreshed to get the updated dictionary metadata. Please note that it’s not necessary to click on the “Don’t Cache” in the upper JCO Destination table if you need to use the Dictionary Invalidation. The above way allows you to select JCO Dictionaries for that destination and invalidate it. Since this runs through a lot of computing, getting the list of dictionaries may not always be best in performance. For performance there is an alternate option where the Dictionary can be invalidated just by typing the name of the dictionary.
The text box expects the name of the dictionary along with the package name. The package name is the name of the model appending with "types". So if there is a ARFC model "com.sap.test.Test", then the dictionary package is "com.sap.test.types.Test".

Examples for the data Dictionary invalidation are like the above where after Short Description of the data element has been changed, the dictionary invalidation is done and refreshing the browser page will reflect the new changed values.

Secondly when an enumeration is changed in the backend for a application, this can immediately reflect after Data Dictionary Cache Invalidation.

Let’s now add a new “Val4” to the Domain on the R/3.
Fig9. The value range is extended for a domain on the R/3 backend
After Dictionary Cache invalidation and browser refresh, this immediately reflects in the application.

Fig10. Updated enumeration values reflecting on the UI

4. Accessing the ARFC Metadata Cache Invalidation Tool
The Metadata Cache Invalidation Tool would be made available from the following releases

1. NW04 or 6.40 SP21 Onwards
2. NW04S or 7.0 SP12 Onwards
3. **NW 7.1 SP3 Onwards.**

This can be accessed through the Web Console application.

![Link to the Web Dynpro console](image)

*Fig11. Link to the Web Dynpro console*

Once there in the left pane click on the “Invalidation of ARFC Metadata Cache” to launch the application.
For older releases please download the following ears from the SDN. Deploy them as normal Web Dynpro applications and run them through the Content Administrator. Here too this supported only from the following releases:

1. NW04 or 6.40 SP20 Onwards
2. NW04s or 7.0 SPS10 Onwards.

The ear files can be found here:

- NW 6.40
- NW 7.0
- NW 7.1

5. Disclaimer

The tool has been intended for ease of use and may not have tested all scenarios. In case some features don't update, please restart the J2EE engine. Any support needed will be provided for the versions mentioned above released with the Web Dynpro Console.

6. Related Content

Please include at least three references to SDN documents or web pages.

1. Adaptive RFC Trouble Shooting Guide
2. How to use the Web Dynpro Content Administrator
3. Adaptive RFC Documentation
4. JCO Quick Summary
7. Acknowledgements

Thanks to Patric Ksinsik for his help in guiding through this topic