

How to Implement BAdI for Enhancing the Datasource in ECC



Applies to:

SAP BW 3.x, BI 7.0 developers and Reporting Users. For more information, visit the [BusinessBusiness Intelligence Home page](#) .

Summary

This document will explain about writing datasource enhancements for Bi extraction in ECC system using BADI. It contains Customized code which helps you to use for any Bi datasource enhancement in ECC with minimal changes.

Author: Suneel Kumar Reddy Sannala

Company: Accenture India Pvt Ltd

Created on: 2nd March 2011

Author Bio

Suneel is a SAP-BI Senior Consultant in Accenture India Pvt LTD, India. His Expertise includes BI reporting and ABAP developments.

Table of Contents

1. Introduction	3
2. Business Scenario	3
3. Step by Step Solution	3
3.1 Create Implementation with the standard BAdI Definition	3
3.2. Implementing the DATA_TRANSFORM method.....	5
3.3. Writing Source code in the DATA_TRANSFORM method	6
3.4. Creating a customized method for each datasource enhancement	7
3.5. Source code in the method M_0FI_AR_4.....	7
4. Conclusion	9
Related Content.....	10
Disclaimer and Liability Notice.....	11

1.Introduction

The SAP enhancement RSAP0001 is used to fill the fields which are added to the extraction structure of the datasource, From Release 6.0, the Business Add-In (BAdI) **RSU5_SAPI_BADI** is available for datasource enhancements. So you will have several advantages while using BAdI instead of User exits.

Note: In User exit, only one enhancement will be used for all the datasources, using BAdI, we can use multiple enhancements. Each enhancement will be implemented in a separate method of the class.

2. Business Scenario

Data source 0FI_AR_4 is appended with fields ZZSPART– Division, ZZVKORG- Sales organization, ZZVTWEG-Distribution channel. The data for these fields should be filled from VBRK table.

3. Step by Step Solution

3.1 Create Implementation with the standard BAdI Definition

Go to T code SE19. Choose Classic BAdI, Give the BAdI name as RSU5_SAPI_BADI (standard). Click on Create Implementation.

The screenshot shows the 'Create Implementation' dialog box. It has two radio buttons: 'New BAdI' and 'Classic BAdI'. The 'Classic BAdI' option is selected. Below it, there is a text field for 'BAdI Name' containing 'RSU5_SAPI_BADI'. At the bottom left, there is a button labeled 'Create Impl.' which is circled in red.

Give the Implementation name and Click on Ok. An Implementation ZC_RSU5_SAPI_BADI will be created with the definition of standard BAdI RSU5_SAPI_BADI.

The screenshot shows the 'Business Add-In Builder: Create Implementation' dialog box. It has two fields: 'Definition Name' with the value 'RSU5_SAPI_BADI' and 'Implementation Name' with the value 'ZC_RSU5_SAPI_BADI'. The 'Implementation Name' field is highlighted with a red box. At the bottom left, there are two icons: a green checkmark and a red X.

Maintain the Description for the implementation and activate it.

Business Add-In Builder: Change Implementation ZC_RSU5_SAPI_BADI

Implementation Name: ZC_RSU5_SAPI_BADI Inactive

Implementation Short Text: Datasource Enhancements

Definition name: RSU5_SAPI_BADI

Runtime Behavior: Implementation will not be called

Attributes Interface

General Data

Package

Implementation of the BAdI is created.

When you create the Implementation, a class will be created with the naming convention ZCL_IM_(implementation name with out Z) in our case, ZCL_IM_C_RSU5_SAPI_BADI.

To display the class and its methods, go to Interface tab and double click on the class name. To display the documentation for the BAdI click on Def. documentation tab.

Business Add-In Builder: Change Implementation ZC_RSU5_SAPI_BADI

Implementation Name: ZC_RSU5_SAPI_BADI Active

Implementation Short Text: Datasource Enhancements

Definition name: RSU5_SAPI_BADI

Runtime Behavior: Implementation is called

Attributes Interface

Interface name: IF_EX_RSU5_SAPI_BADI

Name of implementing class: ZCL_IM_C_RSU5_SAPI_BADI

You will enter into the screen of Class builder

Class Builder: Change Class ZCL_IM_C_RSU5_SAPI_BADI

Class Interface: ZCL_IM_C_RSU5_SAPI_BADI Implemented / Active

Properties Interfaces Friends Attributes Methods Events Types Aliases

Parameters Exceptions

Method	Level	Visibility	M...	Description
IF_EX_RSU5_SAPI_BADI~DATA_TRANSFORM	Instance	Public		Business Add-Ins Method for General Data Transfer
IF_EX_RSU5_SAPI_BADI~HIER_TRANSFORM	Instance	Public		Business Add-Ins Method for Hierarchy Data Transfer

In the class builder, methods tab – we will see 2 standard methods.

3.2. Implementing the DATA_TRANSFORM method

This method allows you to fill your added fields that you have attached as an append structure to the extract structure of transaction and master data in the SAP BW.

To see the parameters passed to this method, select the DATA_TRANSFORM method and click on parameters tab.

Class Builder: Change Class ZCL_IM_C_RSU5_SAPI_BADI

Class Interface: ZCL_IM_C_RSU5_SAPI_BADI | Implemented / Active

Properties | Interfaces | Friends | Attributes | **Methods** | Events | Types | Aliases

Method parameters: DATA_TRANSFORM

Parameter	Type	Pa	O	Typing M	Associated Type	Default value	Description
I_DATASOURCE	Importing	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Type	RSAOT_OLTPSOURCE		DataSource Name
I_UPDMODE	Importing	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Type	SBIWA_S_INTERFACE		Transfer Mode (Full, Delta)
I_T_SELECT	Importing	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Type	SBIWA_T_SELECT		Requested Selection Conditions
I_T_FIELDS	Importing	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Type	SBIWA_T_FIELDS		Requested Fields
C_T_DATA	Changing	<input type="checkbox"/>	<input type="checkbox"/>	Type	ANY TABLE		Extracted Data
C_T_MESSAGES	Changing	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Type	RSU5_T_MESSAGES		Application Log: Error Log

I_DATASOURCE parameter will contain the datasource for transaction and master data and it is a importing parameter to this method. C_T_DATA is a changing parameter, it holds entire for that particular datasource.

Note: Since our current requirement is to fill the data for enhanced transaction datasource, so we will concentrate more on DATA_TRANSFORM method only. To know more about HIER_TRANSFORM method and it's parameters you can refer the SAP note 691154.

To go back to the methods screen just click on the methods tab.

After coming back to methods, double click on the method DATA_TRANSFORM or select the method and click on the source code button, you will go to the source code editor for this method.

Class Builder: Change Class ZCL_IM_C_RSU5_SAPI_BADI

Class Interface: ZCL_IM_C_RSU5_SAPI_BADI | Implemented / Active

Properties | Interfaces | Friends | Attributes | **Methods** | Events | Types

Parameters | Exceptions | **Source Code** | ...

Method	Level	Visibility	M...	Description
IF_EX_RSU5_SAPI_BADI~DATA_TRANSFORM	Instance	Public		Business Add-Ins Meth
IF_EX_RSU5_SAPI_BADI~HIER_TRANSFORM	Instance	Public		Business Add-Ins Meth

Double click on this

You entered into the source code page, here you write the code which should call a method created for each datasource.

3.3. Writing Source code in the DATA_TRANSFORM method

Just copy and paste the below code. This is a common code for any project. You need to change only the class name while reading the SEOCOMPO table. Give the class name you created. Rests of all steps are same as they are.

Note: We wrote this code to use it for any kind of datasource enhancements including CRM or APO. Every line of code, comments are provided to understand the code better.

```
METHOD IF_EX_RSU5_SAPI_BADI~DATA_TRANSFORM.

*Declare the variable which holds method (datasource enhancement)
  DATA : L_METHOD TYPE SEOCMPNAME.

*Add some letter to prefix, because method can't be started with number
  CONCATENATE 'M_' I_DATASOURCE INTO L_METHOD.

*checks the internal table, if it has no data then it exits the method.
  CHECK C_T_DATA[] IS NOT INITIAL.

*Read the component(method) from SEOCOMPO table: this table will get an entry wh
en
* a customized method is created in the class. this customized method will conta
in the source code logic to fill the enhanced field of the datasource

  SELECT SINGLE CMPNAME FROM SEOCOMPO INTO L_METHOD WHERE
    CLSNAME = 'ZCL_IM_C_RSU5_SAPI_BADI' AND
    CMPNAME = L_METHOD.

* Check the sy-subrc, if it is not equal to 0, then exits the method
  CHECK SY-SUBRC EQ 0.

* if method is found in the SEOCOMPO table, that particular method will be call
ed
so automatically enhancement logic written in that method will get executed and
Modified to C_T_DATA

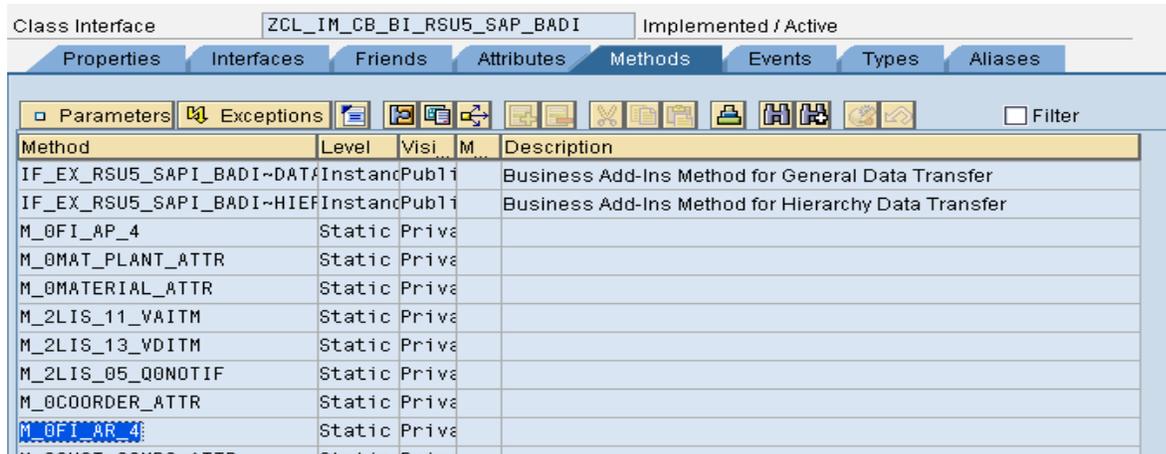
  CALL METHOD (L_METHOD)
    EXPORTING
      I_UPDMODE       = I_UPDMODE
      I_T_SELECT      = I_T_SELECT
      I_T_FIELDS      = I_T_FIELDS
    CHANGING
      C_T_DATA        = C_T_DATA
      C_T_MESSAGES    = C_T_MESSAGES.

ENDMETHOD.
```

Activate this method and come back to the class builder screen.

3.4. Creating a customized method for each datasource enhancement

In our case the datasource is OFI_AR_4.



A new customized method M_OFI_AR_4 is entered in the method's column. Declare this method with level as 'Static' and visibility as 'Private'.

As already discussed, method can't be started with a number, so it was prefixed with M_. The method name we give here should have the same name which is called in DATA_TRANSFORM method.

Note: while declaring this method, we have the two options to choose level, either Static or Instance. If I choose Static - then we can call that method using class name, that method is independent of that object.

If it is instance - then we can call that method using object name, that method is dependent of that object.

Similarly the visibility, we have 3 options to choose, Public, private and protected. If it is Public – then visible to all classes. If it protected – then visible to only with in the class and with in the sub class. If you choose Private – only with in the class, not even from subclass also.

3.5. Source code in the method M_OFI_AR_4

Write your logic to fill the appended fields in the datasource.

Before writing the logic, in the method screen click on private section to declare the parameters like C_T_DATA, I_T_SELECT and I_T_FIELDS.



Copy and paste the below code in the private section of the method. Just replace your method name.

```
private section.
```

```
type-pools SBIWA.
```

```
class-methods M_OFI_AR_4
```

```
importing
```

```
value(I_UPDMODE) type SBIWA_S_INTERFACE-UPDMODE
```

```
value(I_T_SELECT) type SBIWA_T_SELECT
```

```
value(I_T_FIELDS) type SBIWA_T_FIELDS
```

```
changing
```

```
!C_T_DATA type ANY TABLE
```

```
!C_T_MESSAGES type RSU5_T_MESSAGES optional.
```

After writing this code in private section, activate this section and click on back button. You will come back to the source code screen of method.

Here you write your logic to fill the field just like as you write in CMOD enhancements.

Below is the code for my requirement. Comments are provided to understand better. You can write your own logic in the space.

```
METHOD M_OFI_AR_4.

* declaring a field symbol with type of extract structure of OFI_AR_4
FIELD-SYMBOLS: <L_S_DATA> TYPE DTFIAR_3.

* declare a structure with field required from VBRK table
TYPES: BEGIN OF IT_VBRK,
        V_VBELN TYPE VBELN_VF,
        V_VKORG TYPE VKORG,
        V_VTWEG TYPE VTWEG,
        V_SPART TYPE SPART,
      END OF IT_VBRK.

* Declare an internal table and work area with above type
DATA: ZBW_VBRK TYPE STANDARD TABLE OF IT_VBRK,
      L_T_DATA TYPE STANDARD TABLE OF DTFIAR_3, " internal table same as C_T_
DATA
      WA_VBRK TYPE IT_VBRK.

* move the entire content into another internal table
L_T_DATA[] = C_T_DATA[].

IF NOT L_T_DATA IS INITIAL.

* read the fields from VBRK table for all entries of L_T_DATA and put them in
to IT_VBRK
SELECT VBELN VKORG VTWEG SPART
      FROM VBRK INTO TABLE ZBW_VBRK
      FOR ALL ENTRIES IN L_T_DATA
      WHERE VBELN = L_T_DATA-BELNR.

      SORT ZBW_VBRK BY V_VBELN.
      CLEAR WA_VBRK.

      LOOP AT L_T_DATA ASSIGNING <L_S_DATA>.

          READ TABLE ZBW_VBRK INTO WA_VBRK WITH KEY V_VBELN = <L_S_DATA>-
BELNR BINARY SEARCH.

          IF SY-SUBRC = 0.

*update the Sales organization, distribution channel and division
field after reading the Billing document number from IT_VBRK table
          <L_S_DATA>-ZZVKORG = WA_VBRK-V_VKORG.
          <L_S_DATA>-ZZVTWEG = WA_VBRK-V_VTWEG.
          <L_S_DATA>-ZZSPART = WA_VBRK-V_SPART .
          ENDIF.

      ENDLOOP.
ENDIF.
```

```
C_T_DATA[] = L_T_DATA[].
```

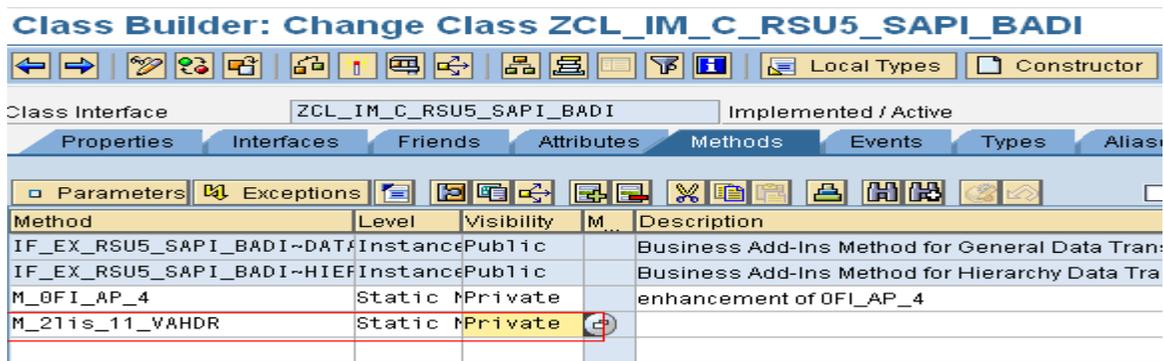
```
REFRESH: L_T_DATA,  
         ZBW_VBRK.
```

```
ENDMETHOD.
```

Save and Activate the method.

So when you extract data in RSA3, datasource parameter is passed to the Class and then respective method will be called from the DATA_TRANSFORM method. That particular method will execute the code and data will be populated for the enhanced fields i.e. ZZSPART– Division, ZZVKORG- Sales organization, ZZVTWEG-Distribution channel in the OFI_AR_4.

Note: To create another enhancement for the datasource 2lis_11_vahdr, give the method name in the class builder and follow the same steps as above.



4. Conclusion

Like this we can write all our enhancements in separate methods in the same class. By this we improve performance and flexibility to work on respective enhancements without disturbing other enhancements.

Related Content

<http://www.sdn.sap.com/irj/scn/go/portal/prtroot/docs/library/uuid/3001894b-b1fb-2910-77ba-e80b6f2053b7?QuickLink=index&overridelayout=true>.

Note 691154 - SAPI with BADI: User exits in SAPI with BADI-interfaces

http://www.sap-hefte.de/download/dateien/1127/098_leseprobe.pdf

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.