Introduction To

BADIs and BADI Implementation

INDEX

BADI (Business Add-Ins)

Purpose of this Document

This document is intended to give a brief introduction of BADI definitions and BADI implementation to the user and also includes a step-by-step guide to create BADI implementation.

Who will benefit?

All technical and functional consultants getting introduced to BADI

Prerequisites

Basic knowledge of User exits.

Scope of this document:

This BOK is mainly focusing on the BADI definition and how to implement the BADI definition. The things I am putting here are exclusively based on my understandings about BADIs.

Introduction

Business Add-Ins or BADI are used to implement customer specific requirements in SAP software. It functions like user exits but are more advance to incorporate the user requirements. BADIs were introduced with the release 4.6.

Concept of BADI

To incorporate the required functionalities in the standard SAP code exit points are defined in this code (Standard SAP code). This achieves the intention of the programmer to add his changes in the standard SAP code without changing the actual code. This is called as "Definition view".

In the "implementation views" programmer chooses his required method depending on the import parameters and adds the code in that method.

Single business add-in contains all the interfaces required to implement specific task.

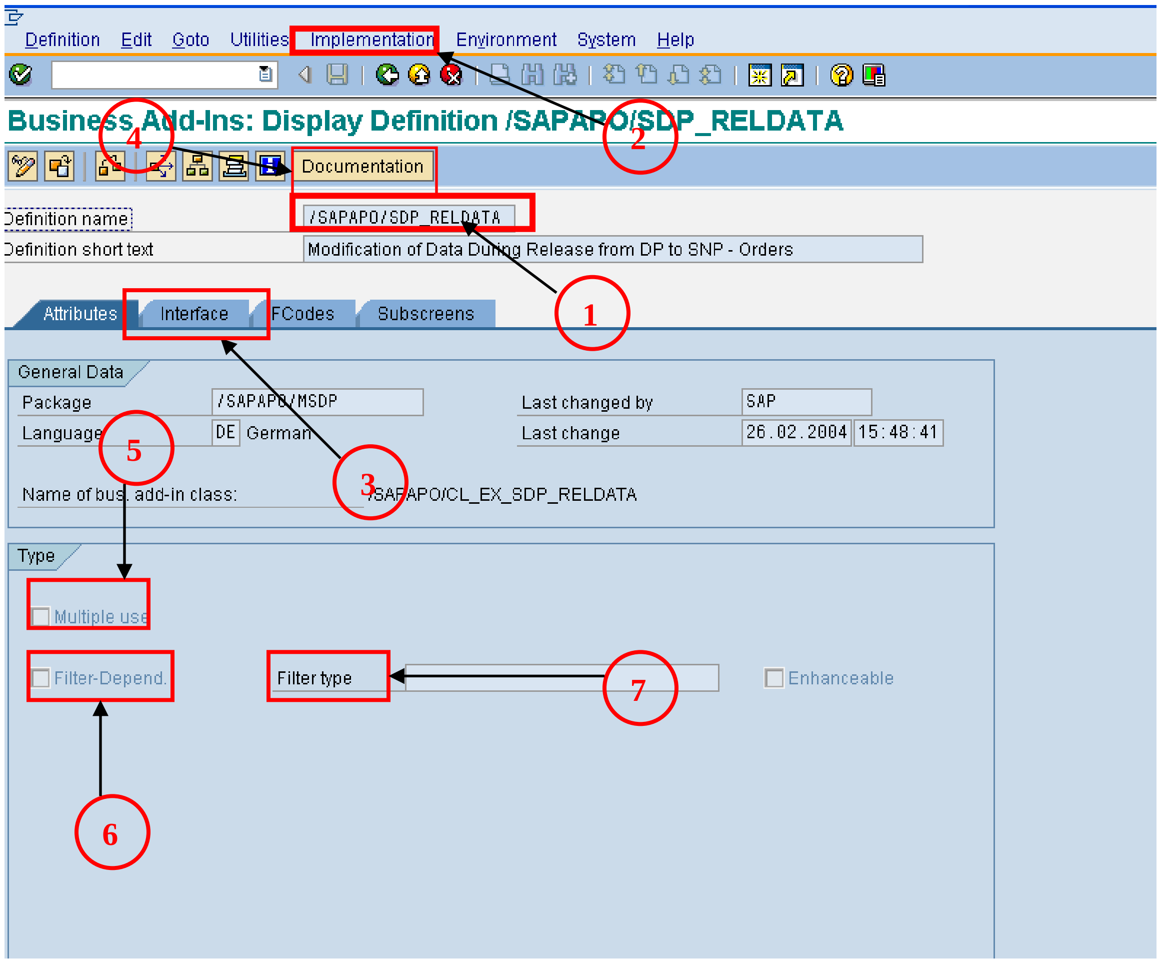
Comparison between BADI and User exits

BADI is an object oriented version of user exit. Unlike in user exit where we write our code in function module, in BADI we enter the code in the methods which are triggered at predefined points by standard SAP program.

What constitutes the BADI definition?

To see the BADI definitions use the transaction SE18.
Or use path choose *Tools → ABAP Workbench → Utilities → Business Add-Ins.*
Take an example of the BADI definition /SAPAPO/SDP_RELDATA.

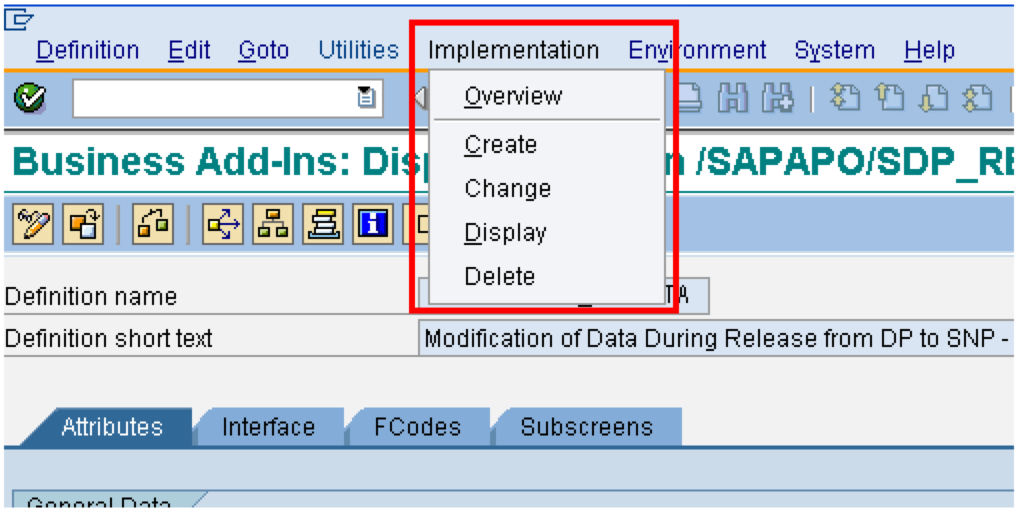
This BADI is used to process the demands from DP to SNP in APO.


Definition Edit Goto Utilities Implementation Environment System Help
Business Add-Ins: Display Definition /SAPAPO/SDP_RELDATA
Documentation
Attributes
Interface
FCodes
Subscreens
General Data
Type
Multiple use
Filter-Depend.
Filter type
Enhanceable
1
2
3
4
5
6
7

Definition name	/SAPAPO/SDP_RELDATA
Definition short text	Modification of Data During Release from DP to SNP - Orders
Package	/SAPAPO/MSDP
Language	DE German
Last changed by	SAP
Last change	26.02.2004 15:48:41
Name of bus. add-in class:	/SAPAPO/CL_EX_SDP_RELDATA

1) The highlighted text shows the BADI definition. To implement the BADI its definition must have been created. The BADI definition could of maximum 20 characters.

2) The implementation drop down menu shows the following items regarding the implementation of BADI definition.



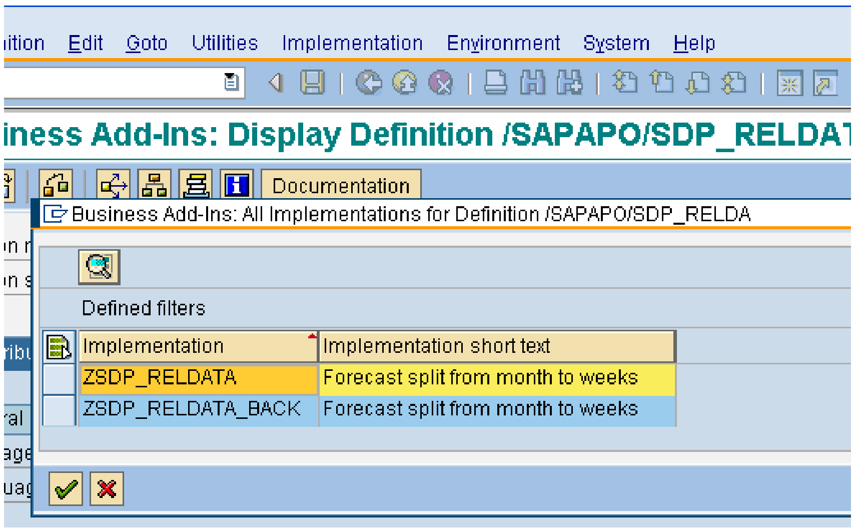
Definition name

Definition short text: Modification of Data During Release from DP to SNP -

a) Create: We can create the new BADI implementation. Implementation name could be of maximum 20 characters.

b) Change: We can make changes in the already implemented BADI definition.

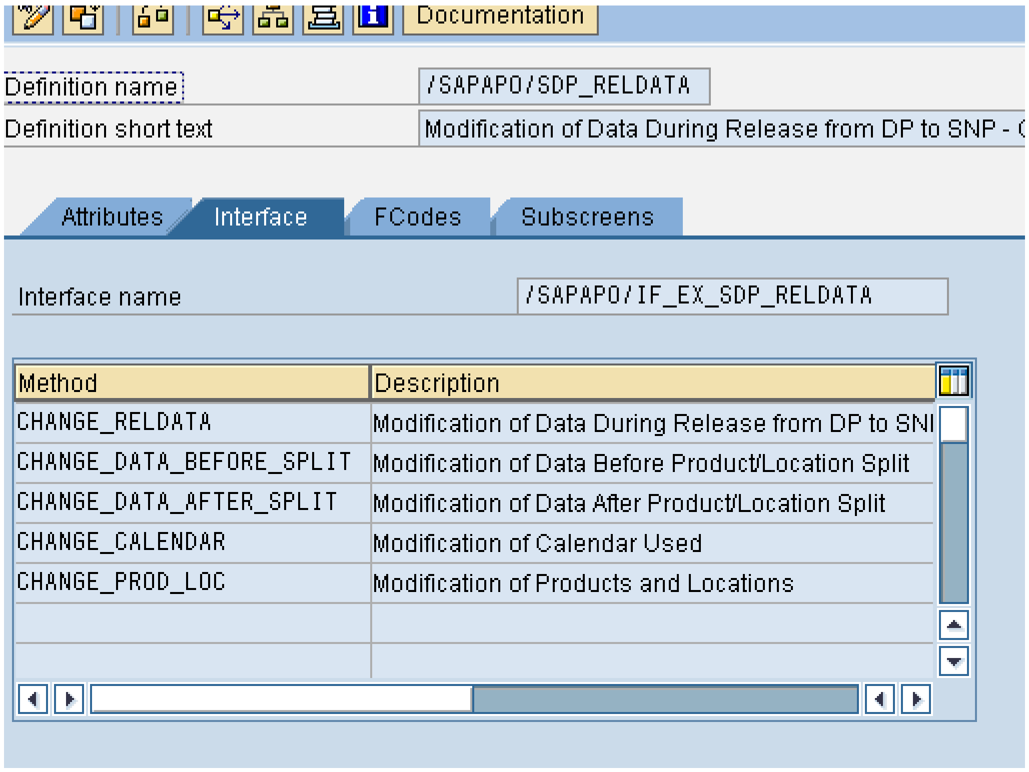
c) Display: It shows the names of the implementation crated for the definition. See the screen shot below:


Edit
Goto
Utilities
Implementation
Environment
System
Help
iness Add-Ins: Display Definition /SAPAPO/SDP_RELDAT
Documentation
Business Add-Ins: All Implementations for Definition /SAPAPO/SDP_RELDA
Defined filters

Implementation	Implementation short text
ZSDP_RELDATA	Forecast split from month to weeks
ZSDP_RELDATA_BACK	Forecast split from month to weeks

Currently there are two implementations created for the definition /SAPAPO/SDP_RELDATA, which are ZSDP_RELDATA and ZSDP_RELDATA_BACK.

3) Interfaces tab :



Definition name	/SAPAPO/SDP_RELDATA
Definition short text	Modification of Data During Release from DP to SNP - (

Interface name /SAPAPO/IF_EX_SDP_RELDATA

Method	Description
CHANGE_RELDATA	Modification of Data During Release from DP to SNI
CHANGE_DATA_BEFORE_SPLIT	Modification of Data Before Product/Location Split
CHANGE_DATA_AFTER_SPLIT	Modification of Data After Product/Location Split
CHANGE_CALENDAR	Modification of Calendar Used
CHANGE_PROD_LOC	Modification of Products and Locations

The interface tab shows the names of the various methods that can be implemented. Each method is having its own functionality.

E.g the method change_reldata when implemented, gives the liberty to programmer to change the quantity and period data which is getting transferred from DP (demand Planning) to SNP (Supply network Planning).

Method CHANGE_DATA_BEFORE_SPLIT comes in to play before modification of data before product/Location split.

4) Documentation: Use the documentation tab to see the documentation for the definition.

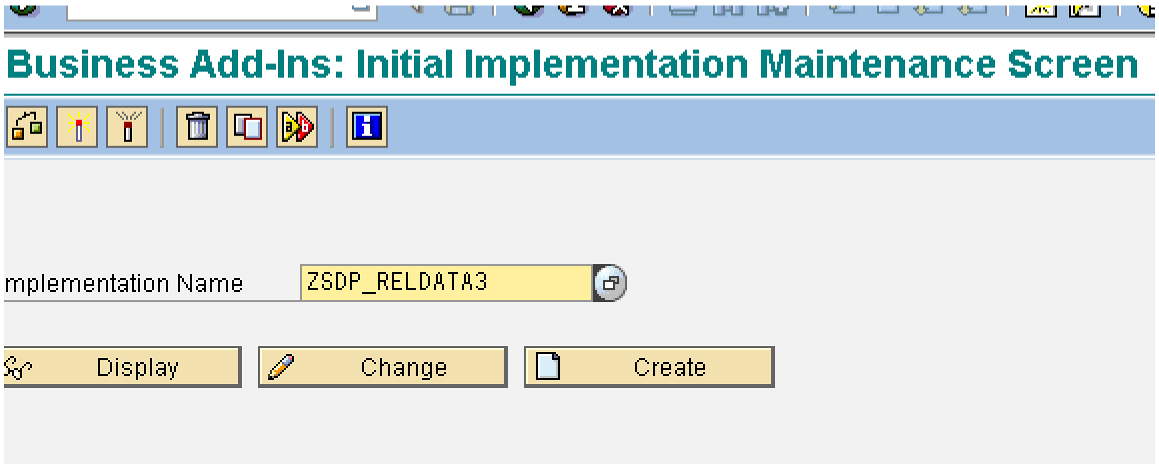
5) Multiple Use Check Box: If this Check is checked, this ensures that the BADI definition can be implemented more than once else it can be implemented only once.

6) Filter-depend and Filter type: This defines the type of data that is valid as a filter value. We can specify the filter type (7), which must be a data element of character type with maximum length as 30.

How to implement the BADI Definition?

To implement BADI definition /SAPAPO/SDP_RELDATA, use implementation name as ZSDP_RELDATA3.

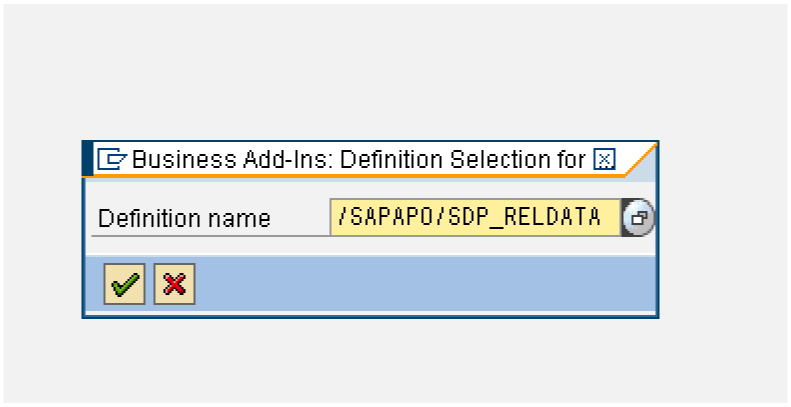
To implement the BADI definitions use transaction SE19.



Business Add-Ins: Initial Implementation Maintenance Screen

mplementation Name ZSDP_RELDATA3

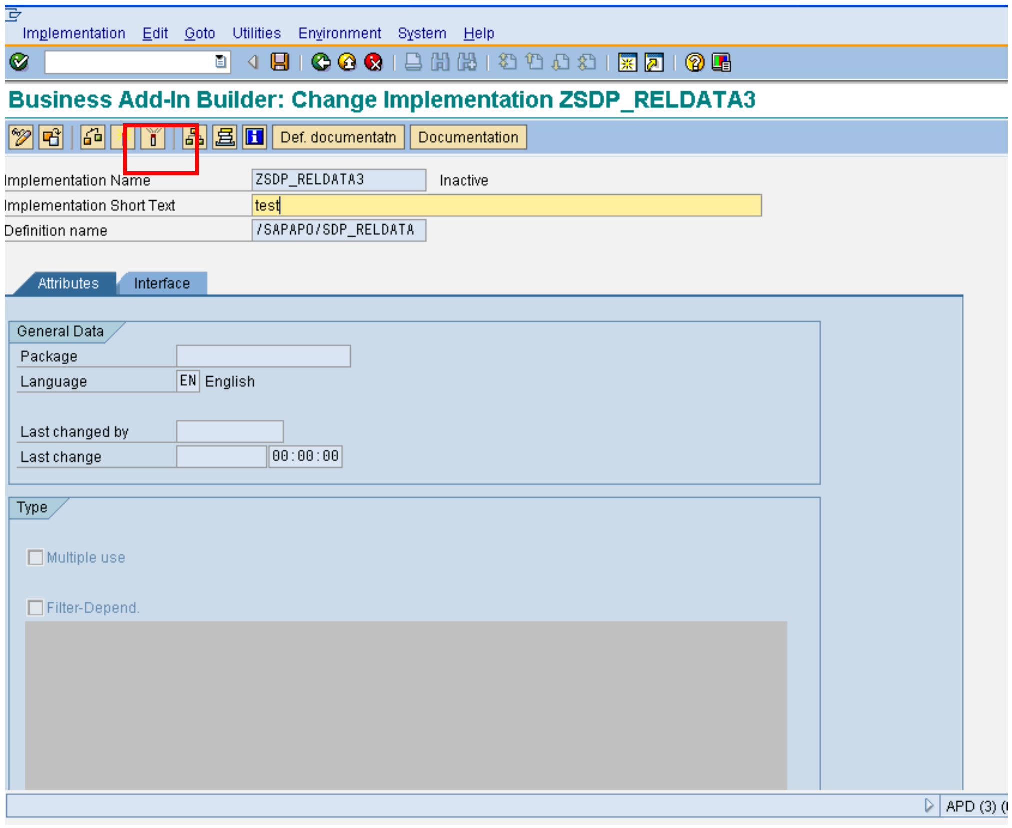
Enter the name of implementation and click on the “create” tab.



Business Add-Ins: Definition Selection for

Definition name /SAPAPO/SDP_RELDATA

Pop-up will appear asking for the definition name, enter the name /SAPAPO/SDP_RELDATA as shown in above screen shot.



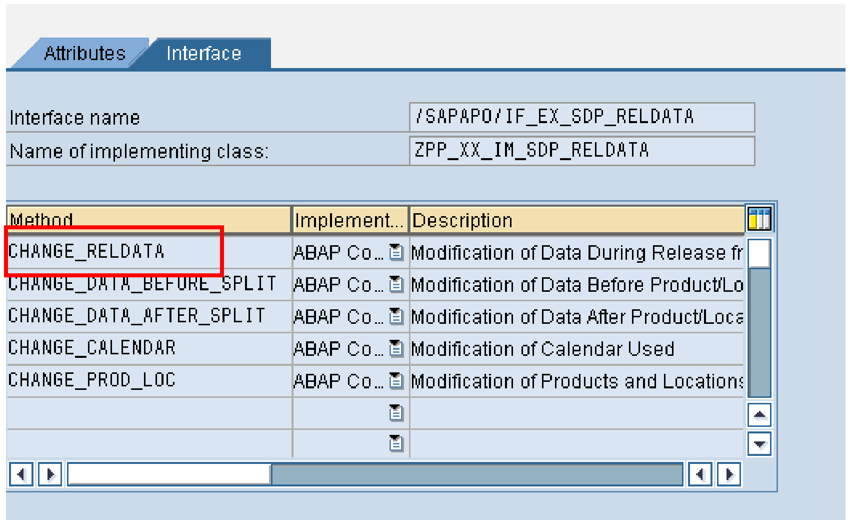
Business Add-In Builder: Change Implementation ZSDP_RELDATA3

Implementation Name	ZSDP_RELDATA3	Inactive
Implementation Short Text	test	
Definition name	/SAPAPO/SDP_RELDATA	
Package		
Language	EN	English
Last changed by		
Last change		00:00:00

Multiple use
Filter-Depend.

Enter the description for the implementation and activate it. To activate and deactivate the implementation use the buttons highlighted in the screen shot.

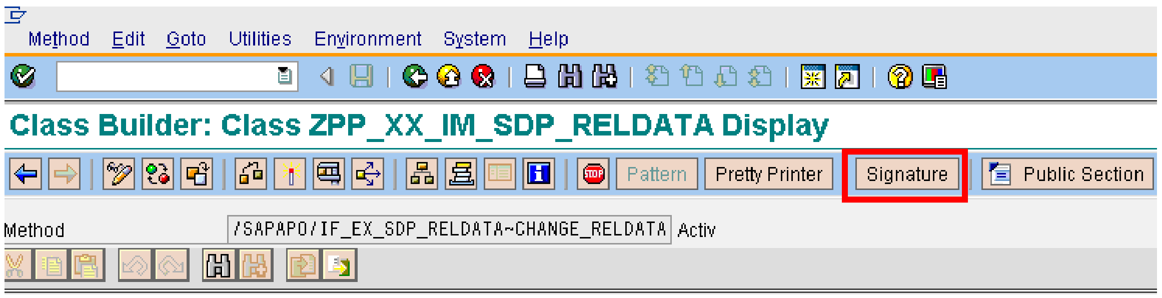
Now to implement the method go to tab interfaces. It will show the names of the methods that can be implemented.



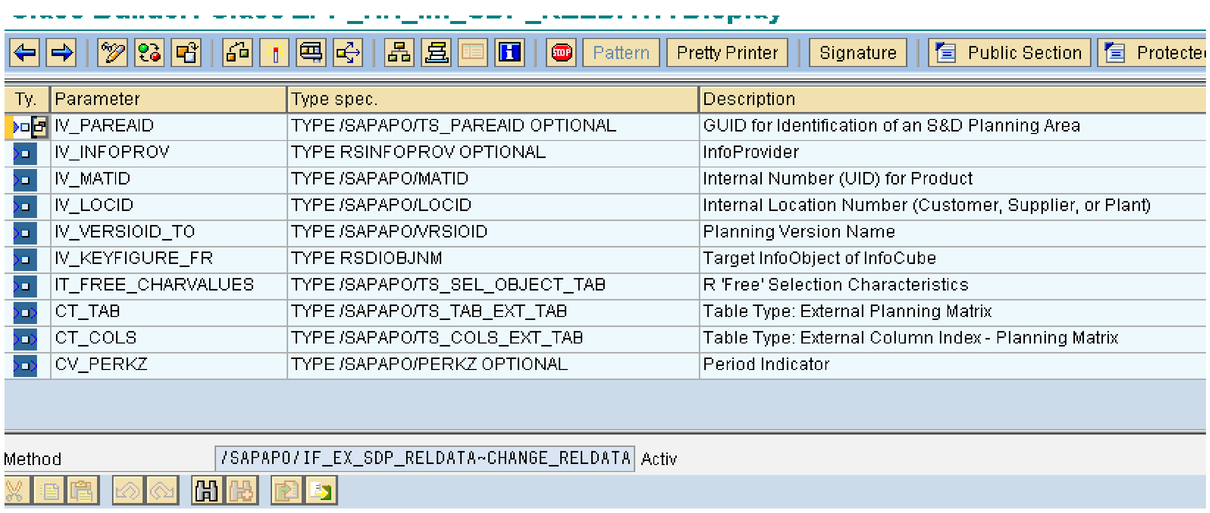
Interface name	/SAPAPO/IF_EX_SDP_RELDATA
Name of implementing class:	ZPP_XX_IM_SDP_RELDATA

Method	Implement...	Description
CHANGE_RELDATA	ABAP Co...	Modification of Data During Release fr
CHANGE_DATA_BEFORE_SPLIT	ABAP Co...	Modification of Data Before Product/Lo
CHANGE_DATA_AFTER_SPLIT	ABAP Co...	Modification of Data After Product/Loca
CHANGE_CALENDAR	ABAP Co...	Modification of Calendar Used
CHANGE_PROD_LOC	ABAP Co...	Modification of Products and Locations

Our required method is change_reldata. To write the code in to the method double click on it.


Method Edit Goto Utilities Environment System Help
Class Builder: Class ZPP_XX_IM_SDP_RELDATA Display
Pattern
Pretty Printer
Signature
Public Section
Method /SAPAPO/IF_EX_SDP_RELDATA~CHANGE_RELDATA Activ

The editor will appear in where we can write our code.
To see the import parameters click on the "Signature" tab as highlighted above.



Ty.	Parameter	Type spec.	Description
	IV_PAREAID	TYPE /SAPAPO/TS_PAREAID OPTIONAL	GUID for Identification of an S&D Planning Area
	IV_INFOPROV	TYPE RSINFOPROV OPTIONAL	InfoProvider
	IV_MATID	TYPE /SAPAPO/MATID	Internal Number (UID) for Product
	IV_LOCID	TYPE /SAPAPO/LOCID	Internal Location Number (Customer, Supplier, or Plant)
	IV_VERSIOID_TO	TYPE /SAPAPO/VRSIOID	Planning Version Name
	IV_KEYFIGURE_FR	TYPE RSDIOBJNM	Target InfoObject of InfoCube
	IT_FREE_CHARVALUES	TYPE /SAPAPO/TS_SEL_OBJECT_TAB	R 'Free' Selection Characteristics
	CT_TAB	TYPE /SAPAPO/TS_TAB_EXT_TAB	Table Type: External Planning Matrix
	CT_COLS	TYPE /SAPAPO/TS_COLS_EXT_TAB	Table Type: External Column Index - Planning Matrix
	CV_PERKZ	TYPE /SAPAPO/PERKZ OPTIONAL	Period Indicator

Here as we can see parameters are starting with IV, IT and CT. The meaning of this is as follows:

1) Starting with IV: Parameter name starting with IV is a variable which is being imported in to the method with some value.

2) Starting with IT: Parameter starting with IT is a name of internal table which is being imported in to the method with lines of records. We can use these internal tables only for read purpose; we can not make any changes to the values of it.

3) Starting with CT: Parameter starting with CT is a name of internal table which is being imported in to the method with lines of record. We can make the changes to values of the data in these tables.

Again clicking on the "Signature" tab disappears the parameter pane.

How BADI Works?

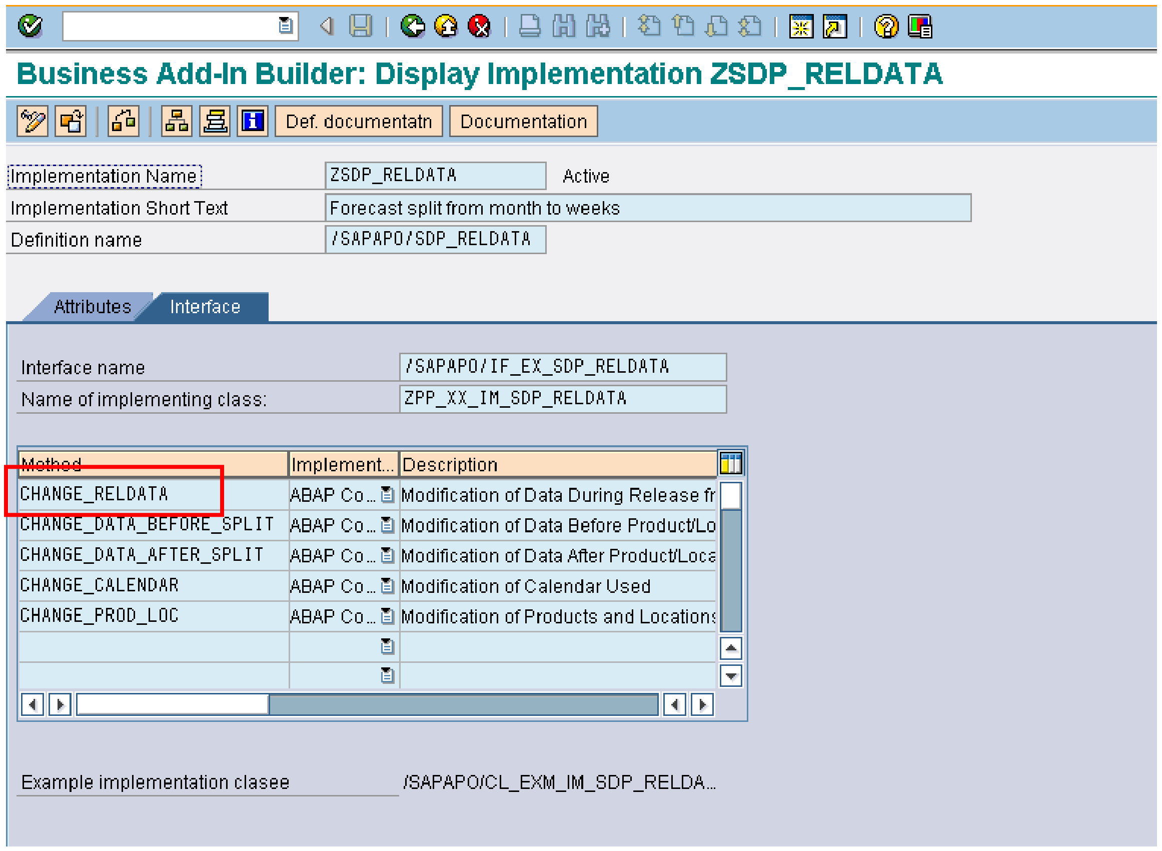
As Mentioned earlier functionality of BADIs and User exits is same. Only difference is BADI uses object oriented approach to achieve the functionality of the user exit. Both are the exits of the standard SAP program.

When we say BADI is being called, actually methods are being called which contain our code.

Let me take you through an example:

This example is taken from APO Rollout Project.
Here we used the transaction /SAPAPO/MC90 which transfers the Demand Planning data in to Supply network Planning.

When this transaction is executed a standard SAP program is called which subsequently calls the BADI implementation method of the BADI implementation ZSDP_RELDATA. This is the implementation of definition /SAPAPO/SDP_RELDATA.


Business Add-In Builder: Display Implementation ZSDP_RELDATA

Implementation Name	ZSDP_RELDATA	Active
Implementation Short Text	Forecast split from month to weeks	
Definition name	/SAPAPO/SDP_RELDATA	

Interface name	/SAPAPO/IF_EX_SDP_RELDATA
Name of implementing class:	ZPP_XX_IM_SDP_RELDATA

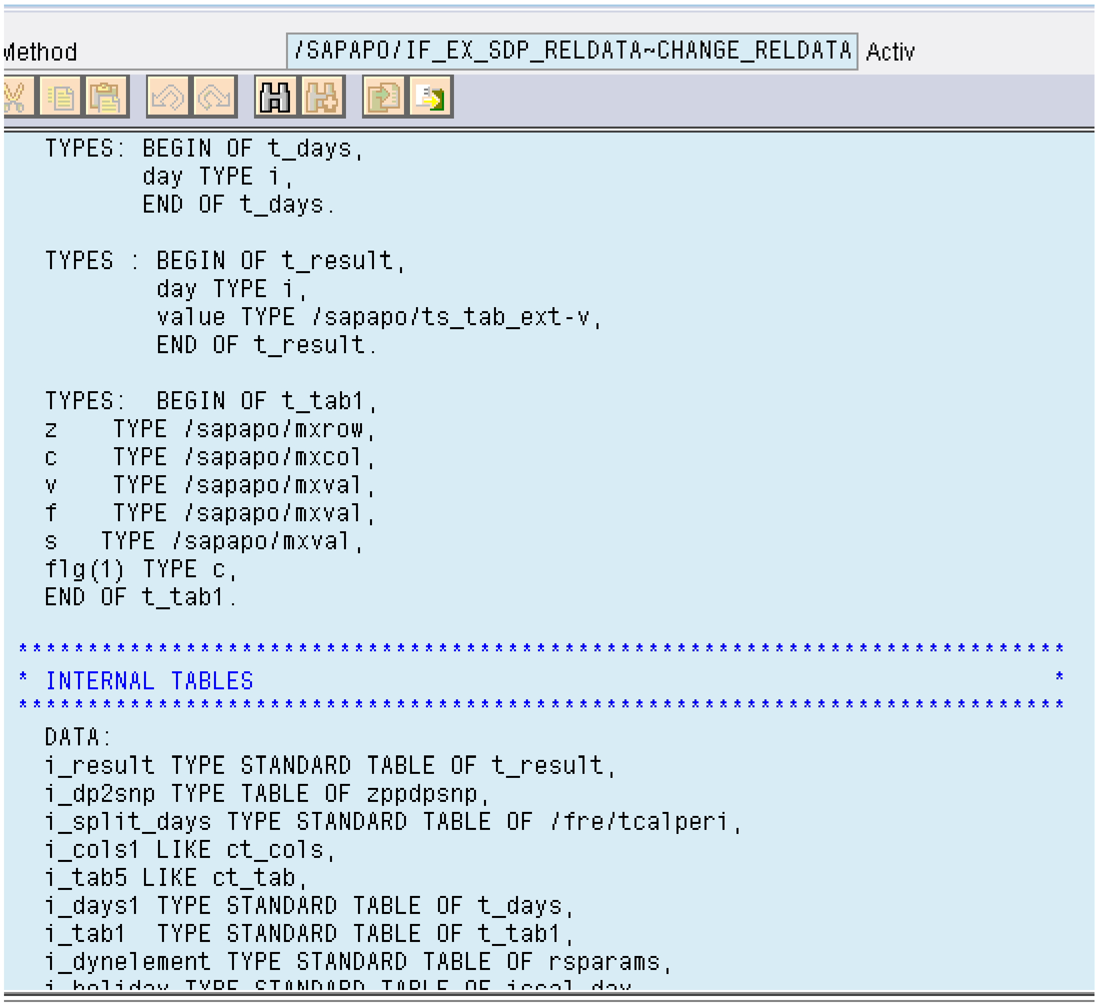
Method	Implement...	Description
CHANGE_RELDATA	ABAP Co...	Modification of Data During Release fr
CHANGE_DATA_BEFORE_SPLIT	ABAP Co...	Modification of Data Before Product/Lo
CHANGE_DATA_AFTER_SPLIT	ABAP Co...	Modification of Data After Product/Loca
CHANGE_CALENDAR	ABAP Co...	Modification of Calendar Used
CHANGE_PROD_LOC	ABAP Co...	Modification of Products and Locations

Example implementation clasee /SAPAPO/CL_EXM_IM_SDP_RELDA...

Our aim was to split the data as per the requirement, for that we used the method CHANGE_RELDATA. The above screen shot shows this method.

The code was been written in this method to get the desired data split.

Code in the method.



```
  TYPES: BEGIN OF t_days,
         day TYPE i,
         END OF t_days.

  TYPES : BEGIN OF t_result,
          day TYPE i,
          value TYPE /sapapo/ts_tab_ext-v,
          END OF t_result.

  TYPES:  BEGIN OF t_tab1,
  z    TYPE /sapapo/mxrow,
  c    TYPE /sapapo/mxcol,
  v    TYPE /sapapo/mxval,
  f    TYPE /sapapo/mxval,
  s   TYPE /sapapo/mxval,
  flg(1) TYPE c,
  END OF t_tab1.

*************************************************************************
* INTERNAL TABLES                                                      *
*************************************************************************
  DATA:
  i_result TYPE STANDARD TABLE OF t_result,
  i_dp2snp TYPE TABLE OF zppdpsnp,
  i_split_days TYPE STANDARD TABLE OF /fre/tcalperi,
  i_cols1 LIKE ct_cols,
  i_tab5 LIKE ct_tab,
  i_days1 TYPE STANDARD TABLE OF t_days,
  i_tab1  TYPE STANDARD TABLE OF t_tab1,
  i_dynelement TYPE STANDARD TABLE OF rsparams,
```

By changing the values in the import parameters we achieved the desired split.