

# How To... Dynamically Call DataSource Specific Programs in Source System User Exits

Applicable Releases:

SAP NetWeaver BI 3.x and 7.0

IT Practice:

Business Information Management

IT Scenario:

Enterprise Data Warehouse

Version 1.1

September 2008

© 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, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, Informix, i5/OS, POWER, 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.

SAP NetWeaver "How-to" Guides are intended to simplify the product implementation. While specific product features and procedures typically are explained in a practical business context, it is not implied that those features and procedures are the only approach in solving a specific business problem using SAP NetWeaver. Should you wish to receive additional information, clarification or support, please refer to SAP Consulting.

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.

#### Disclaimer

Some components of this product are based on Java™. Any code change in these components may cause unpredictable and severe malfunctions and is therefore expressly prohibited, as is any decompilation of these components.

Any Java™ Source Code delivered with this product is only to be used by SAP's Support Services and may not be modified or altered in any way.

## Document History

<b>Document Version</b>	<b>Description</b>
1.10	Conversion / Adaption of Document from 3.x to 7.0
1.00	First official release of this guide

## Typographic Conventions

Type Style	Description
<i>Example Text</i>	Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options.  Cross-references to other documentation
<b>Example text</b>	Emphasized words or phrases in body text, graphic titles, and table titles
Example text	File and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
<b>Example text</b>	User entry texts. These are words or characters that you enter in the system exactly as they appear in the documentation.
< <b>Example text</b> >	Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.
EXAMPLE TEXT	Keys on the keyboard, for example, F2 or ENTER.

## Icons





Icon	Description
	Caution
	Note or Important
	Example
	Recommendation or Tip

Table of Contents

- 1. **Business Scenario**..... 1
- 2. **Background Information**..... 1
- 3. **Step-by-Step Procedure**..... 3
- 4. **Appendix** ..... 6
  - 4.1 ZXRSAU02 Code – Form Routine Call ..... 6
  - 4.2 ZBWI\_0MATERIAL\_ATTR Code..... 7
  - 4.3 ZXRSAU02 Code – General Approach ..... 8

## 1. Business Scenario

SAP delivered source system user-exits allow us to incorporate additional custom code typically used to validate, filter or populate customer defined fields in DataSource. For example consider master data user exit ZXRSAU02 (EXIT\_SAPLRSAP\_002). Commonly practiced usage consists of following two approaches:

1. CASE I\_DATASOURCE.  
    WHEN '0MATERIAL\_ATTR'.  
    \*           Custom code to validate, filter or populate fields  
                ...  
    ENDCASE.
2. CASE I\_DATESOURCE.  
    WHEN '0MATERIAL\_ATTR'.  
        INCLUDE ZBWMATERIAL. (This include contains custom code)  
    ENDCASE.

(Please refer to APPENDIX for the sample code)

With both of these approaches transport management can be challenging especially when you have large number of DataSources and different people or teams are working on the user-exits. Object may be locked by someone else while you are trying to incorporate an urgent fix to resolve a production issue. This leads to reversal of code and retesting of the extractors which are redundant and unproductive.

## 2. Background Information

Simple ABAP technique using PERFORM or CALL FUNCTION statement will resolve this issue permanently by calling the program related to the DataSource dynamically.

### Syntax:

```
PERFORM form_name  
    IN PROGRAM (dynamically created program name)  
    TABLES table1 table2 ...  
    IF FOUND.
```

Name of the form "form\_name" must be same across all DataSource includes. A naming convention with appropriate prefix has to be followed for this technique to work. Prefix "ZBWI" is used in the sample code and customer can adapt other naming conventions as necessary.

**Benefits:**

- ✓ Elimination of transport management issues with respect to user-exits
- ✓ User-Exit Includes are only locked by people responsible for it
- ✓ You will never have to change the user-exits such as ZXRSAU01, ZXRSAU02 etc

 Tip

You can implement the same for all user-exits (Transactional, Text and Hierarchy) but make sure to correctly program the interface to the form routine by using appropriate tables as part of the perform statement.

### 3. Step-by-Step Procedure

The code demonstrated here is only applicable to Master Data user-exit. You can use the same principle for other user-exits.

1. Identify the user exit you want to customize. Shown here is EXIT\_SAPLRSAP\_002 for Master Data.

```

FUNCTION EXIT_SAPLRSAP_002.
*-----
**"Lokale Schnittstelle:
**  IMPORTING
**      VALUE(I_DATASOURCE) TYPE  RSAOT_OLTPSOURCE
**      VALUE(I_CHABASNM) TYPE  SBIWA_S_INTERFACE-CHABASNM
**      VALUE(I_UPDMODE) TYPE  SBIWA_S_INTERFACE-UPDMODE
**
**  TABLES
**      I_T_SELECT TYPE  SBIWA_T_SELECT
**      I_T_FIELDS TYPE  SBIWA_T_FIELDS
**      I_T_DATA
**      I_C_T_MESSAGES STRUCTURE  BALMI OPTIONAL
**
**  EXCEPTIONS
**      RSAP_CUSTOMER_EXIT_ERROR
*-----

INCLUDE ZXRSAU02.

ENDFUNCTION.

```

2. Change the code in ZXRSAU02 to the given code

Please note that prefix for the program is hard coded here and you may adapt different standards as you see fit naming standards for your organization.

"IF FOUND" option in the perform statement will ignore if there is no user-exit "includes" associated with a DataSource.

```

*&-----
*&  INCLUDE ZXRSAU02
*&-----
*
*local variable for the include/program name for each datasource
DATA: l_prog_name LIKE trdir-name.

*-- ZBWI_ is the assumed prefix for the datasource program

CONCATENATE 'ZBWI_' i_datasource INTO l_prog_name.

*-- execute corresponding user exit. if exist
PERFORM execute_user_exit IN PROGRAM (l_prog_name)
  TABLES I_T_SELECT
           I_T_FIELDS
           I_T_DATA
           I_T_MESSAGES
           IF FOUND.

```



- Write the program specific to DataSource. Here 0MATERIAL\_ATTR is enhanced and program ZBWI\_0MATERIAL\_ATTR is created for the user exit.

Here you can see C\_T\_DATA is already type casted with DataSource structure and you don't need locally define a structure to map each record

```

REPORT ZBWI_0MATERIAL_ATTR.
*--- User exit program for datasource 0MATERIAL_ATTR
TYPE-POOLS: sbiwa. "Mandatory for typecasting

*&-----*
*&      Form start_user_exit
*&-----*
FORM execute_user_exit
  TABLES I_T_SELECT  TYPE      SBIWA_T_SELECT
           I_T_FIELDS TYPE      SBIWA_T_FIELDS
           C_T_DATA   STRUCTURE BIW_MARA_S " Datasource structure
           C_T_MESSAGES STRUCTURE BALMI.

  *-- table definition here          (if any) -----*
  *-- local data definition here    (if any) -----*

  loop at c_t_data.
  *-- get Assortment list type from table MARA
  select single BBTYP
    into c_t_data-zzbbtyp
    from MARA
    where MATNR = c_t_data-matnr.
    if sy-subrc = 0.
      modify c_t_data.
    endif.
  endloop.

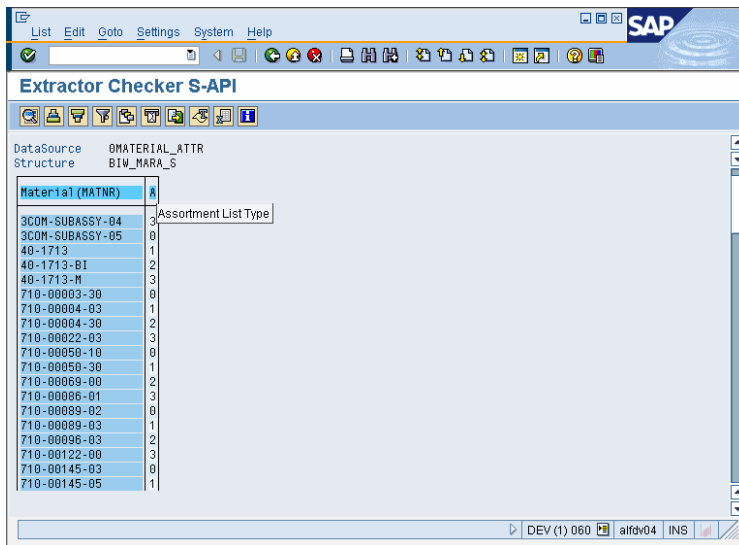
ENDFORM.          " start_user_exit
    
```

- Make sure to change the attribute type of your include/program to Executable program. This is mandatory for this technique to work.

ABAP: Program Attributes ZBWI\_0MATERIAL\_ATTR Change

Title	Include program for datasource 0MATERIAL_ATTR	
Original language	EN	English
Created	04/18/2006	ERANEZHD
Last changed by	04/18/2006	ERANEZHD
Status	Active	
<b>Attributes</b>		
Type	Executable program	
Status	Customer Production Program	

5. Activate program ZBWI\_0MATERIAL\_ATTR and test your extractor using transaction RSA3 (Extract Checker)



 Note

For new DataSource user-exit customization, copy the program and reuse it with appropriate changes.

Same principle can be applied to all other DataSource user-exits (Transactional Data, Text and Hierarchy); for that matter any where you need to incorporate different includes/programs by passing same data.

## 4. Appendix

### Appendix A – Source Code

Code	Description
ZXRSAU02	Sample Master Data user-exit with FORM routine in report
ZBWI_0MATERIAL_ATTR	Report for user-exit implementation
ZXRSAU02	General Approach

### 4.1 ZXRSAU02 Code – Form Routine Call

```
*&-----
*&  INCLUDE ZXRSAU02
*&-----

*local variable for the include/program name for each datasource
DATA: l_prog_name LIKE trdir-name.

*-- ZBWI_ is the assumed prefix for the datasource program

CONCATENATE 'ZBWI_' i_datasource INTO l_prog_name.

*-- execute corresponding user exit. if exist
PERFORM execute_user_exit IN PROGRAM (l_prog_name)
    TABLES I_T_SELECT
            I_T_FIELDS
            I_T_DATA
            I_T_MESSAGES
    IF FOUND.
```

## 4.2 ZBWI\_0MATERIAL\_ATTR Code

```
REPORT ZBWI_0MATERIAL_ATTR.

*--- User exit program for datasource 0MATERIAL_ATTR
TYPE-POOLS: sbiwa. "Mandatory for typecasting

*&-----*
*&      Form start_user_exit
*&-----*

FORM execute_user_exit
      TABLES I_T_SELECT      TYPE      SBIWA_T_SELECT
              I_T_FIELDS     TYPE      SBIWA_T_FIELDS
              C_T_DATA       STRUCTURE BIW_MARA_S    " Datasource structure
              C_T_MESSAGES  STRUCTURE BALMI.

*-- table definition here          (if any) -----*

*-- local data definition here    (if any) -----*

      loop at c_t_data.

*-- get Assortment list type from table MARA
      select single BBTYP
              into c_t_data-zzbbtyp
              from MARA
              where MATNR = c_t_data-matnr.
      if sy-subrc = 0.
              modify c_t_data.
      endif.
endloop.

ENDFORM.          " start_user_exit
```

## 4.3 ZXRSAU02 Code – General Approach

```

*--- table defintion here      (if any) -----*

*--- local data definition here (if any) -----*
data: lr_biw_mara_s like BIW_MARA_S,
      lr_biw_marc_s like BIW_MARC_S.

CASE i_datasource.
  WHEN '0MATERIAL_ATTR'.
    loop at c_t_data into lr_biw_mara_s.
*-- get Assortment List Type from table Mara
    select single BBTYP
      into lr_biw_mara_s-zzbbtyp
      from MARA
      where matnr = lr_biw_mara_s-matnr.
    if sy-subrc = 0.
      modify c_t_data from lr_biw_mara_s.
    endif.
  endloop.

  WHEN '0MAT_PLANT_ATTR'.
    loop at c_t_data into lr_biw_marc_s.
*-- get Control code for consumption taxes in foreign trade from MARC
    select single STEUC
      into lr_biw_marc_s-zzsteuc
      from MARC
      where matnr = lr_biw_marc_s-matnr
        and werks = lr_biw_marc_s-werks.
    if sy-subrc = 0.
      modify c_t_data from lr_biw_marc_s.
    endif.
  endloop.
  WHEN OTHERS.
ENDCASE.

```

### INCLUDE approach

```

CASE i_datasource.
  WHEN '0MATERIAL_ATTR'.
*-- include for datasource 0MATERIAL_ATTR
    INCLUDE ZBWI_0MATERIAL_ATTR.

  WHEN '0MAT_PLANT_ATTR'.
*-- include for datasource 0MAT_PLANT_ATTR
    INCLUDE ZBWI_0MAT_PLANT_ATTR.

  WHEN OTHERS.
ENDCASE.

```

[www.sdn.sap.com/irj/sdn/howtoguides](http://www.sdn.sap.com/irj/sdn/howtoguides)