

How to Log Organization Attribute Changes in mySAP SRM



Applies to:

MySAP SRM 5.5. For more information, please visit [SCN SRM Home Page](#).

Summary

The article proposes a simplified solution for logging the attribute changes that are carried out through the Maintain Attributes Transaction (PPOMA_BBP). The solution uses a standard BADI to create a log (SLG1) containing the details of the old and new attribute values of the Org Unit / Position which is modified.

Thus the system administrator can monitor all changes and ensure data integrity. This is more of a prototype/reusable application, please feel free to adopt it your specific requirements.

There are so many SDN SRM Forum threads which discuss the need for a solution similar to this, hope this is helpful.

Author: Kathirvel Balakrishnan

Company: Wipro Technologies

Created on: 10 June 2009

Author Bio



Kathirvel Balakrishnan is a SAP certified SRM Consultant with Wipro Technologies and has nearly 5 years of experience with SAP.

Table of Contents

Introduction	3
Application Log: Object Maintenance (Transaction SLG0)	3
Implementing BADI - HRBAS00INFTY.....	4
Source Code: Method IF_EX_HRBAS00INFTY~IN_UPDATE	4
Solution Results.....	8
Disclaimer and Liability Notice.....	10

Introduction

SRM Organization structure is a hierarchical representation of the various organization units according to the task and functions. An organizational unit represents any type of organizational entity found within a company, for example, subsidiaries, divisions, departments, or special project teams. Organizational units are one of the objects that make up organizational plans. Once the complete structure is set-up users are assigned to these units against a valid position.

Apart from this they also need to be assigned the attributes which play a crucial in controlling the tasks or functions that the user or organization unit can perform. The organization unit creation, user assignment, attributes maintenance, etc are day to day activities in the procurement system. But careless or incorrect attribute assignment or deletion can cause serious issues; a few include:

- a. Access to procure for organization for which they are not allowed
- b. Display/Access to procurement documents of other organization units
- c. Access to different backend system
- d. Improper deletion of attributes can prevent users from carrying out their day to day procurement activities

This is a simplified prototype which generates a log (SLG1) every time a change is introduced in the organization structure (PPOMA_BBP). Thus system administrators can monitor the attribute changes and ensure data integrity. This also helps in identifying the old values that were modified.

This is more of a prototype/reusable application, please feel free to adopt it your specific requirements.

Note: The SRM version used for developing this example is SRM 5.5; there could be additional changes for other SRM versions for achieving this functionality.

Application Log: Object Maintenance (Transaction SLG0)

The first step is to create a new application log object for capturing the organization attribute changes. A new object called ZPPOMA_BBP is to be created for this example.



Implementing BADI - HRBAS00INFTY

The next step is to implement the BADI HRBAS00INFTY for capturing the changes as logs. A new implementation named ZBBP_PPOMA_CHG_LOG is created as shown below. The implementation is to be saved and activated. The method IN_UPDATE will contain the logic for capturing the attributes changes made and create a log with these details.

Business Add-In Builder: Display Implementation ZBBP_PPOMA_CHG_LOG

Implementation Name: ZBBP_PPOMA_CHG_LOG Active

Implementation Short Text: Change Log Generation for SRM Organization Structure

Definition name: HRBAS00INFTY

Attributes Interface

Interface name: IF_EX_HRBAS00INFTY

Name of implementing class: ZCL_IM_BBP_PPOMA_CHG_LOG

Method	Implement...	Description
BEFORE_OUTPUT	ABAP ABAF	Call-Up Time PBO
AFTER_INPUT	ABAP ABAF	Call-Up Time PAI
BEFORE_UPDATE	ABAP ABAF	Call-Up Before Update
IN_UPDATE	ABAP ABAF	Call-Up During Update

Source Code: Method IF_EX_HRBAS00INFTY~IN_UPDATE

The complete source code for this method is given below. Just copy and paste the same to your implementation. The code is very much self-explanatory.

The method IN_UPDATE of this BADI is called just before saving the attribute changes to the database. The logic is to generate a log with the old and new attributes.

```

METHOD if_ex_hrbas00infty-in_update.
* -----*
*OLD_IMAGE      TYPE BEF_IMAGE_TAB
*NEW_IMAGE      TYPE AFT_IMAGE_TAB
*PLOG_TAB       TYPE HRDBTAB_TAB
*TB_PLOG_TAB    TYPE HRTBUFFER_TAB
* -----*

* Declaration for Local Tables
DATA: lt_log_handle TYPE bal_t_logh.
DATA: lt_hrt1222    TYPE TABLE OF hrt1222.

* Declaration for Local Structures
DATA: ls_plog_tab  TYPE hrdbtabs.
DATA: ls_s_log     TYPE bal_s_log.
DATA: ls_tb_plog_tab TYPE hrtbuffer.
DATA: ls_hrt1222  TYPE hrt1222.

* Declaration for Local Variables
DATA: lv_log_handle TYPE balloghndl.
DATA: lv_text(1000) TYPE c.
DATA: lv_tabnr      TYPE hrtabnr.

* Proceed only if data is available
IF NOT old_image IS INITIAL AND NOT tb_plog_tab IS INITIAL.

* Create persistent application log handler
CLEAR: ls_s_log, lv_text.
ls_s_log-object = 'ZPPOMA_BBP'.
ls_s_log-aluser  = sy-uname.
ls_s_log-alprog  = sy-repid.
ls_s_log-aldate  = sy-datlo.
ls_s_log-althme = sy-uzeit.
ls_s_log-altcode = sy-tcode.
ls_s_log-extnumber = 'ZPPOMA_BBP_CHANGE_LOG'.

CALL FUNCTION 'BAL_LOG_CREATE'
  EXPORTING
    i_s_log          = ls_s_log
  IMPORTING
    e_log_handle    = lv_log_handle
  EXCEPTIONS
    log_header_inconsistent = 1
    OTHERS              = 2.
IF sy-subrc <> 0.
  MESSAGE ID sy-msgid
           TYPE sy-msgty
           NUMBER sy-msgno
           WITH sy-msgv1
                sy-msgv2
                sy-msgv3
                sy-msgv4.
ENDIF.

```

```

* Get the Key for the entry that is being deleted
  READ TABLE plog_tab INTO ls_plog_tab
    WITH KEY opera = 'D'.
  IF sy-subrc EQ 0.
*   Read the existing values from DB
  lv_tabnr = ls_plog_tab-vdata.
  SELECT * FROM hrt1222
    INTO TABLE lt_hrt1222
      WHERE tabnr = lv_tabnr.
  IF sy-subrc EQ 0.
* Adding a header for Old Attribute Values
  CLEAR lv_text.
  lv_text = 'List of Old Attribute Values'(001).
  CALL FUNCTION 'BAL_LOG_MSG_ADD_FREE_TEXT'
    EXPORTING
      i_log_handle      = lv_log_handle
      i_msgty           = 'S'
      i_probclass      = '1'
      i_text            = lv_text
    EXCEPTIONS
      log_not_found    = 1
      msg_inconsistent = 2
      log_is_full      = 3
      OTHERS           = 4.
  IF sy-subrc <> 0.
    MESSAGE ID sy-msgid
      TYPE sy-msgty
      NUMBER sy-msgno
      WITH sy-msgv1
          sy-msgv2
          sy-msgv3
          sy-msgv4.
  ENDIF.

LOOP AT lt_hrt1222 INTO ls_hrt1222 .
  CLEAR lv_text.
  CONCATENATE ls_hrt1222-attrb      ls_hrt1222-low
              ls_hrt1222-high      ls_hrt1222-excluded
              ls_hrt1222-defaultval ls_hrt1222-inherited
              INTO lv_text RESPECTING BLANKS.
  CALL FUNCTION 'BAL_LOG_MSG_ADD_FREE_TEXT'
    EXPORTING
      i_log_handle      = lv_log_handle
      i_msgty           = 'W'
      i_probclass      = '1'
      i_text            = lv_text
    EXCEPTIONS
      log_not_found    = 1
      msg_inconsistent = 2
      log_is_full      = 3
      OTHERS           = 4.
  IF sy-subrc <> 0.
    MESSAGE ID sy-msgid
      TYPE sy-msgty
      NUMBER sy-msgno
      WITH sy-msgv1

```

```

                                sy-msgv2
                                sy-msgv3
                                sy-msgv4.
    ENDIF.
  ENDLOOP.
ENDIF.
ENDIF.

* Adding a header for New Attribute Values
lv_text = 'List of New Attribute Values'(002).
CALL FUNCTION 'BAL_LOG_MSG_ADD_FREE_TEXT'
  EXPORTING
    i_log_handle      = lv_log_handle
    i_msgty           = 'S'
    i_probclass       = '1'
    i_text            = lv_text
  EXCEPTIONS
    log_not_found     = 1
    msg_inconsistent = 2
    log_is_full       = 3
    OTHERS            = 4.
IF sy-subrc <> 0.
  MESSAGE ID sy-msgid
           TYPE sy-msgty
           NUMBER sy-msgno
           WITH sy-msgv1
                sy-msgv2
                sy-msgv3
                sy-msgv4.
ENDIF.

LOOP AT tb_plog_tab INTO ls_tb_plog_tab .
  CLEAR lv_text.
  lv_text = ls_tb_plog_tab-tdata.
  CALL FUNCTION 'BAL_LOG_MSG_ADD_FREE_TEXT'
    EXPORTING
      i_log_handle      = lv_log_handle
      i_msgty           = 'W'
      i_probclass       = '1'
      i_text            = lv_text
    EXCEPTIONS
      log_not_found     = 1
      msg_inconsistent = 2
      log_is_full       = 3
      OTHERS            = 4.
  IF sy-subrc <> 0.
    MESSAGE ID sy-msgid
             TYPE sy-msgty
             NUMBER sy-msgno
             WITH sy-msgv1
                  sy-msgv2
                  sy-msgv3
                  sy-msgv4.
  ENDIF.
ENDLOOP.

```

```
* Finally save the log entries
APPEND lv_log_handle TO lt_log_handle.
CALL FUNCTION 'BAL_DB_SAVE'
  EXPORTING
    i_in_update_task = 'X'
    i_save_all       = ' '
    i_t_log_handle   = lt_log_handle
  EXCEPTIONS
    log_not_found    = 1
    save_not_allowed = 2
    numbering_error  = 3
    OTHERS           = 4.
IF sy-subrc <> 0.
  MESSAGE ID sy-msgid
           TYPE sy-msgty
           NUMBER sy-msgno
           WITH sy-msgv1
                sy-msgv2
                sy-msgv3
                sy-msgv4.
ENDIF.

ENDIF.
ENDMETHOD.
```

Solution Results

Once the above activities have been completed; just launch the transaction PPOMA_BBP and perform some attribute change for an organizational unit or a user. The BADI will get triggered and the old and new attribute values will be captured.

Then log can be viewed in the transaction SLG1. The results of this solution are attached below:

SLG1 Selection Screen

Analyse Application Log

Change Log Generation for SRM Organization Structur
 Subobject
 External ID

Time Restriction
 From (Date/Time)
 To (Date/Time)

Log Triggered By
 User
 Transaction code
 Program

SLG1 Output Screen (with attributes)

Display logs

Date/Time/User	Numb	External ID	Object txt	Sub-object text	Tran	Program	Mode	Log number
08.06.2009 14:37:57 BKATHIRVI	28	ZPPOMA_BBP_...	Change Log Ge...		PPOMA_B...	ZCL_IM_B...	Dialog pro...	0000000000000679358
08.06.2009 14:45:07 BKATHIRVI	29	ZPPOMA_BBP_...	Change Log Ge...		PPOMA_B...	ZCL_IM_B...	Dialog pro...	0000000000000679368
08.06.2009 14:58:48 BKATHIRVI	58	ZPPOMA_BBP_...	Change Log Ge...		PPOMA_B...	ZCL_IM_B...	Dialog pro...	0000000000000679369
08.06.2009 15:02:16 BKATHIRVI	59	ZPPOMA_BBP_...	Change Log Ge...		PPOMA_B...	ZCL_IM_B...	Dialog pro...	0000000000000679371

Ty...	Message Text
■	List of Old Attribute Values
▲	CNT DR0CLNT4001BN001
▲	CUR GBP
▲	BWA DR0CLNT4001201
▲	BWA DR0CLNT4001261
▲	BWA DR0CLNT4001281
▲	CT_PROC_TYPCTR
▲	DP_PROC_TYEC
▲	PS_ORCOD 12301
▲	FORWARD_WIX
▲	BPO DR0CLNT4001CBAMMD1 1 08 08

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.