How To...

Master Data Governance for Material:

Send a Mail notification during the governance process

Applicable Releases:

EhP5

Version 1.1

November 2011
### Document History

<table>
<thead>
<tr>
<th>Document Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>First official release of this guide</td>
</tr>
</tbody>
</table>
## Typographic Conventions

<table>
<thead>
<tr>
<th>Type Style</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example Text</strong></td>
<td>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</td>
</tr>
<tr>
<td><strong>Example text</strong></td>
<td>Emphasized words or phrases in body text, graphic titles, and table titles</td>
</tr>
<tr>
<td><strong>Example text</strong></td>
<td>File and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.</td>
</tr>
<tr>
<td><strong>Example text</strong></td>
<td>User entry texts. These are words or characters that you enter in the system exactly as they appear in the documentation.</td>
</tr>
<tr>
<td><strong>&lt;Example text&gt;</strong></td>
<td>Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.</td>
</tr>
<tr>
<td><strong>EXAMPLE TEXT</strong></td>
<td>Keys on the keyboard, for example, F2 or ENTER.</td>
</tr>
</tbody>
</table>

## Icons

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>🔴</td>
<td>Caution</td>
</tr>
<tr>
<td>🔭</td>
<td>Note or Important</td>
</tr>
<tr>
<td>📖</td>
<td>Example</td>
</tr>
<tr>
<td>🚨</td>
<td>Recommendation or Tip</td>
</tr>
</tbody>
</table>
# Table of Contents

1. Business Scenario ................................................................................................................. 1

2. Background Information ........................................................................................................ 1

3. Step-by-Step Procedure ......................................................................................................... 2
   3.1 Email notification with subworkflow ............................................................................... 2
       3.1.1 Create Subworkflow ............................................................................................ 2
       3.1.2 Customizing BRF+ Tables for CR-Type EM_MAT05 ........................................... 5
       3.1.3 Process ................................................................................................................ 6
   3.2 Email notification with BADI system call method .......................................................... 6
       3.2.1 Create service name in IMG ............................................................................... 7
       3.2.2 Enhancement spot USMD_SSW_SERVICE_PROCESSOR ..................................... 7
       3.2.3 Customizing BRF+ Tables .................................................................................. 15

4. Additional Information .......................................................................................................... 16
   4.1 Workitem Notification via Email .................................................................................... 16
1. Business Scenario

SAP Master Data Governance, material data (MDGM) provides business processes to find, create, change, and delete material master data. It supports the governance of material master data in a central hub and the distribution to connected operational and business intelligence systems.

The processes are workflow-driven and can include several approval and revision phases, and the collaboration of all users participating in the master data maintenance.

This scenario addresses if the customer requires to notify a person via Email (for example the requestor) when the workflow is in a certain step or completed.

This How To Guide describes possible solution to send out Email notification from the Rules Based Workflow.

2. Background Information

There are different possibilities. A subworkflow for Email notification can be included in the Rules Based Workflow, also a call for a system method. All of this definition occurs within the workflow configuration.
3. Step-by-Step Procedure

3.1 Email notification with subworkflow

The following is a description of how to send an Email notification from the MDG Rules Based Workflow WS60800086 with a subworkflow.

Example is in Q35/Client 002.

3.1.1 Create Subworkflow

SWDD:
How To - Master Data Governance for Material: Send a Mail Notification During the Governance Process

Send a Mail notification
How To - Master Data Governance for Material: Send a Mail Notification During the Governance Process

Send a Mail notification

Activate your subworkflow.
3.1.2 Customizing BRF+ Tables for CR-Type EM_MAT05

3.1.2.1 Create new CR-Type

3.1.2.2 Single Value Decision Table
3.1.2.3 Non User Agent Decision Table

3.1.3 Process

After activation there a mail will be issued to the specified user. You can check with transaction SOST:

3.2 Email notification with BADI system call method

The following is a description on how to send an Email notification to from the MDG workflow WS60800086 with a BADI system call method. It is not meant to enable notification of workitem. That can be handled via SAP Extended Notification. This is for non-workitem notifications.

It is a first attempt in which some pieces are leveraged from other examples of code. That additional work can be done to create email types, define related parameters and store message text in one of SAP's text tables.

Example is in PL2/Client 820.
3.2.1 Create service name in IMG

<table>
<thead>
<tr>
<th>Service Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WS9896242</td>
<td>Subworkflow to Notify Requestor</td>
</tr>
<tr>
<td>WS9896243</td>
<td>Subworkflow to Display Material</td>
</tr>
<tr>
<td>ZCOMPLETED_MERCHANT</td>
<td>Completed Merchant Processing</td>
</tr>
<tr>
<td>ZDETERMINE_MERCHANT</td>
<td>Determine Merchant</td>
</tr>
<tr>
<td>ZNOTIFY_REQUESOR</td>
<td>System Call to Send Mail</td>
</tr>
<tr>
<td>ZUPDATE_PARENT_SELLSTATUS</td>
<td>Update Parent Sellstatus Field</td>
</tr>
</tbody>
</table>

3.2.2 Enhancement spot USMD_SSW_SERVICE_PROCESSOR

Create a BADI implementation for definition USMD_SSW_SYSTEM_METHODCALLER and use service name as filter for implementation

In implementing class of BADI (here it is ZCL_CT_SYS_METHOD_CALLER)
Example coding:

```
METHOD if_usmd_ssw_syst_method_caller~call_system_method.

DATA:
  lo_crequest TYPE REF TO if_usmd_crequest_api,
  lo_crequest_zflavor TYPE REF TO if_usmd_crequest_api,
  lo_crequest_zparent TYPE REF TO if_usmd_crequest_api,
  lo_cr_api_instance TYPE REF TO if_usmd_crequest_api,
  lo_instance TYPE REF TO if_usmd_model,
  lo_instance_ext TYPE REF TO if_usmd_model_ext,
  lv_sydatum TYPE usmd_txtlg,
  lv_syzeit TYPE usmd_txtlg,
  lv_crequest_id TYPE usmd_crequest,
  lv_crequest_text TYPE usmd_txtlg,
  lv_crequest_type TYPE usmd_crequest_type,
  lv_reason TYPE usmd_reason,
  lv_parent_id TYPE zparent,
  lv_mm_model TYPE usmd_model VALUE 'MM',
  lv_mm_zparent TYPE usmd_entity VALUE 'ZPARENT',
  lv_mm_zflavor TYPE usmd_entity VALUE 'MATERIAL',
  lv_zparent_fieldname TYPE usmd_fieldname VALUE 'ZPARENT',
  lv_zflavor_fieldname TYPE usmd_fieldname VALUE 'MATERIAL',
  lt_crequest_entity TYPE usmd_t_crequest_entity,
  ls_crequest_entity TYPE usmd_s_crequest_entity,
  lt_added_entity TYPE usmd_t_crequest_entity,
```

```
lr_data_structure  TYPE REF TO data,
lr_data_table     TYPE REF TO data,
lr_zpar_data_structure TYPE REF TO data,
lr_zpar_data_table  TYPE REF TO data,

lr_zmdgflv_data_structure TYPE REF TO data,
lr_zmdgflv_data_table   TYPE REF TO data,

lv_entity_fieldname  TYPE usmd_fieldname,
lv_flv_zsellsta      TYPE zm_sell_status,
lv_par_zsellsta      TYPE zm_sell_status,
lv_zparent           TYPE zparent,

lt_entities          TYPE usmd_t_crequest_entity,
ls_entity            TYPE usmd_s_crequest_entity,

lt_message           TYPE usmd_t_message,
lt_crequest          TYPE STANDARD TABLE OF usmd120c,
ls_crequest          TYPE usmd_s_crequest,
ls_status_txttable   TYPE usmd130t,
lr_data              TYPE REF TO data,
lr_table             TYPE REF TO data,
lst                     TYPE usmd_ts_sel,
ls_sel                TYPE usmd_s_sel.

FIELD-SYMBOLS:  <lt_mat_int> TYPE ANY TABLE,
                <ld_mat_in> TYPE any,
                <ls_material> TYPE any,
                <lv_subfamily> TYPE any,
                <lv_zonline> TYPE any,
                <lv_zretail> TYPE any,
                <lv_value> TYPE any.
FIELD-SYMBOLS:
                <structure> TYPE any,
                <table> TYPE INDEX TABLE,
                <field> TYPE any.

CASE iv_service_name.

WHEN 'ZNOTIFY_REQUESTOR'.

* Get and read CR and launch email notification

CALL METHOD cl_usmd_crequest_api=>get_instance
    EXPORTING
        iv_crequest     = iv_cr_number
    IMPORTING
        re_inst_crequest_api = lo_crequest.

CALL METHOD lo_crequest->read_crequest
    IMPORTING

How To - Master Data Governance for Material: Send a Mail Notification During the Governance Process

Send a Mail notification

```
es_crequest = ls_crequest.

SELECT SINGLE * INTO ls_status_txttable FROM usmd130t
WHERE langu = 'EN'
AND usmd_creq_status = ls_crequest-usmd_creq_status.

CALL FUNCTION 'ZCT_USER_SEND_MAIL'
EXPORTING
  iv_crequest_id = iv_cr_number
  iv_user_name = ls_crequest-usmd_created_by
  iv_status_text = ls_status_txttable-txtmi.
```

Badi calls Function Module logic to build text and send out as email. Example coding:

```
FUNCTION zct_user_send_mail.
**"*"Local Interface:
**"*" IMPORTING
**"*" REFERENCE(IV_CREQUEST_ID) TYPE USMD_CREQUEST DEFAULT '000000000204'
**"*" REFERENCE(IV_USER_NAME) TYPE SYUNAME DEFAULT 'I010123'
**"*" REFERENCE(IV_STATUS_TEXT) TYPE USMD_TXTMI
**"*" Example Report for usage of CL_BCS_MESSAGE: RSBCS_EXAMPLE_EMAIL
**"*" DATA:
lo_message TYPE REF TO cl_bcs_message,
lx_bcs_send TYPE REF TO cx_bcs_send,
lv_err_text TYPE string,

lv_crequest_ext TYPE char12,
lv_status_text TYPE usmd_txtmi,
lv_mail_address TYPE string,
lv_creator_name TYPE ad_namtext,
lv_subject TYPE string,
lv_body TYPE string.

**"*" PERFORM init USING
PERFORM init USING iv_crequest_id
PERFORM init USING iv_user_name
PERFORM init USING iv_status_text

CHANGING lv_crequest_ext
  lv_status_text
  lv_mail_address
  lv_creator_name.
```
IF lv_mail_address IS NOT INITIAL.

TRY.
  CREATE OBJECT lo_message.

*** create subject
  PERFORM create_subject USING lv_crequest_ext
    CHANGING lv_subject.
  lo_message->set_subject( lv_subject ).

*** create body as single text document
  PERFORM create_body USING lv_crequest_ext
    lv_status_text
    lv_creator_name
    CHANGING lv_body.
  lo_message->set_main_doc( lv_body ).

*** set recipient
  lo_message->add_recipient( lv_mail_address ).

*** set attributes
  lo_message->set_status_mail( 'N' ).
  lo_message->set_requested_status('N').
  lo_message->set_importance('N').
  lo_message->set_send_immediately('X').

*** send mail
  lo_message->send( ).
  lo_message->show_send_log( ).

  CATCH cx_bcs_send INTO lx_bcs_send.
  lv_err_text = lx_bcs_send->get_text( ).
  MESSAGE lv_err_text TYPE 'E'.
ENDTRY.

ENDIF.

ENDFUNCTION.

/*-----------------------------------------------*/
/* Form create_subject */
/*-----------------------------------------------*/
FORM create_subject USING iv_crequest_id TYPE char12
    CHANGING ev_subject TYPE string.
  CONCATENATE 'Change Request' iv_crequest_id 'has been completed'
    INTO ev_subject SEPARATED BY space.
ENDFORM.

/*-----------------------------------------------*/
Form convert_crequest_id
*---------------------------------------------------------------------*
FORM convert_crequest_id USING iv_crequest_id TYPE usmd_request
    CHANGING ev_crequest_ext TYPE char12.

DATA:
   lo_model TYPE REF TO if_usmd_model,
   lt_message TYPE usmd_t_message.

*---------------------------------------------------------------------*
cl_usmd_model=>get_instance(
   EXPORTING
      i_usmd_model = space
   IMPORTING
      eo_instance = lo_model
      et_message = lt_message ).

LOOP AT lt_message TRANSPORTING NO FIELDS
   WHERE msgty CA usmd0_cs_msgty-severe.
ENDLOOP.
CHECK NOT sy-subrc IS INITIAL.

lo_model->convert_char_output(
   EXPORTING
      i_fieldname = usmd0_cs_fld-crequest
      i_value_int = iv_crequest_id
   IMPORTING
      e_value_ext = ev_crequest_ext ).

ENDFORM.

Form create_body
*---------------------------------------------------------------------*
FORM create_body USING iv_crequest_id TYPE char12
   iv_status_text TYPE usmd_txtmi
   iv_cr_creator TYPE ad_namtext
    CHANGING ev_body TYPE string.

CONSTANTS:
   lc_linefeed TYPE char2 VALUE cl_abap_char_utilities=>cr_lf,
   lc_empty_line TYPE string VALUE space.

DATA:
   lv_cr_creator TYPE string,
   lv_line1 TYPE string,
   lv_line2 TYPE string,
   lv_url TYPE string,
   lv_line3 TYPE string,
   lv_line4 TYPE string,
lv_line5   TYPE string,
lv_body    TYPE string.

*--------------------------------------------------------------------*

lv_line1 = 'Hi,'.
lv_line2 = 'Your change request for creating a material has been complete
d.'.
* CONCATENATE 'https://uxcic5t.wdf.sap.corp.:44371/sap/bc/webdynpro/sap/us
md_entity_value2?SAP-CLIENT=195&SAP-LANGUAGE=EN&IS_CREQ_MODE=X&SAP-WD-
CONFIGID=MDG_MM_APP_BS_MAT_GEN_BAT&CREQUEST=' iv_crequest_id INTO lv_url.
CONCATENATE 'Change Request ' iv_crequest_id ' has status: ' iv_status_t
ext INTO lv_line3 separated by space.
lv_line4 = 'Warm Regards,'.
lv_line5 = 'Apple MDG Administrator'.

CONCATENATE lv_line1
   lc_empty_line
   lv_line2
   *
   lv_url
   lc_empty_line
   lv_line3
   lc_empty_line
   lv_line4
   lv_line5
   INTO ev_body SEPARATED BY lc_linefeed.

ENDFORM.    "create_body

*--------------------------------------------------------------------*

DATA:
  ls_user_data   TYPE bapiaddr3,
  lt_return      TYPE bapirettab,
  ls_crequest_data TYPE usmd120c.

*--------------------------------------------------------------------*

ev_status_text = iv_status_text.
**PERFORM** convert_crequest_id USING iv_crequest_id
    CHANGING ev_crequest_ext.

* get email address user name from provided user id

CALL FUNCTION 'BAPI_USER_GET_DETAIL'
    EXPORTING
        username = iv_user_name
        cache_results = space
    IMPORTING
        address = ls_user_data
    TABLES
        return = lt_return.

ev_mail_address = ls_user_data-e_mail.

* get data of the change request creator

PERFORM get_crequest_data USING iv_crequest_id
    CHANGING ls_crequest_data.

CLEAR ls_user_data.

CALL FUNCTION 'BAPI_USER_GET_DETAIL'
    EXPORTING
        username = ls_crequest_data-usmd_created_by
        cache_results = space
    IMPORTING
        address = ls_user_data
    TABLES
        return = lt_return.

ev_creator_name = ls_user_data-fullname.

ENDFORM.

"init"

§---------------------------------------------------------------------
§ Form get_crequest_data
§---------------------------------------------------------------------
FORM get_crequest_data USING iv_crequest_id TYPE usmd_crequest
    CHANGING ev_crequest_data TYPE usmd120c.

SELECT SINGLE * FROM usmd120c INTO ev_crequest_data
    WHERE
        usmd_crequest = iv_crequest_id.

ENDFORM.

"get_crequest_data"
3.2.3 Customizing BRF+ Tables

3.2.3.1 Create new CR-Type

3.2.3.2 Single Value Decision Table

After successful activation of change request (CR Previous step 91 and previous action 31) the notification step is called.

It identifies next step as 50 which points to the end of the workflow, Condition Alias 99 and New CR Step 99.

3.2.3.3 Non User Agent Decision Table

Using tcode, USMD_SSW_RULE, insert call to method in appropriate location. Here the notification is sent to the creator of the workflow upon successful activation.
Create an entry for a step in Non-user agent decision table for process pattern 02 (synchronous method call)
4. Additional Information

4.1 Workitem Notification via Email

Notifications for SAP Business Workflow notify users by e-mail or SMS about their work items that need to be processed. Notifications are sent to the relevant user in the form of messages. At present, e-mail messages (HTML or text-only) and SMS messages are supported.

Extended notifications are used to deliver email notifications to the agents of a workitem that has been sent to them and is waiting for them to process. It is a Workitem notification and has links/information about the workitem. This notification is generated external to the workflow template.

In a business scenario, for example you could configure extended notifications so that all approving agents get an email telling them that they have a workitem in their inbox (POWL/UWL). It is a notification of a dialog step that they must take action on. Once the Extended Notification mail is sent out the workflow step is still in process waiting for the agent to execute the dialog work item.
How To - Master Data Governance for Material: Send a Mail Notification During the Governance Process

Send a Mail notification

The complete presentation can be found:

Online help: