SAP CRM & SAP Solution Manager
PPF Action Condition Enhancement
Business Object Repository Enhancement

How to add attributes for PPF action conditions?

28.10.2016

Peter Weigel
Hyazinthenstr. 6
D-06122 Halle / Saale

Phone: +49 170 5337567
E-Mail: peter.weigel@hybrid-eichhoernchen.de
Web: www.hybrid-eichhoernchen.de

Content

1 Introduction

1.1 Abstract

The PPF action framework is a powerful framework to trigger actions (= sending smartforms or executing methods) dependent on several events & issues (= schedule and start conditions).

However in SAP standard there is only a small set of business transaction related attributes which can be evaluated. This document explains how to enhance the PPF by additional customer or SAP standard attributes with minimum development effort.

1.2 Example Requirements

- Evaluation customer fields of ORDERADM_H, CUSTOMER_H, ACTIVITIY_H, SERVICE_H or SRV_REQ_H.
- Evaluating whether the field content is changed now (compare field value on database against field value in API memory).
- Evaluating whether some texts resp. text types are maintained.
- Evaluating error messages of business transactions.

1.3 Solution

The data root of the PPF Schedule and Start Conditions is the BOR object of the corresponding Business Transaction. For Change Request Management BOR object BUS2000116 is used. This development solution enhances this BOR object by some additional attributes.

1.4 Side Effect

The attribute enhancements are available for every application using the BOR object BUS2000116. For example SAP Business Workflows can use these attributes too.

It might be possible that the BOR object will be initialized completely when it gets instantiated. Because additional attributes needs additional read logic, the instantiation may needs more time. At the moment it is unknown whether this is really the fact.

1.5 Support Package Upgrade

This development solution contains some Business Object Enhancement (additional attributes for business object BUS2000116). This would lead to problems if the SAP decides to deliver attributes with the same name. In this case our business object ZUS2000116 would have syntax errors and we would need to remove or rename our attribute. After that we need to adjust all dependent schedule and start conditions using these renamed attributes.

1.6 Version History

Version	Author	Date	Comment
1	Peter Weigel	17.06.2015	First Version
2	Peter Weigel	28.10.2016	Evaluation of Error Messages.

1.7 Literature, Disclaimer, Contact and Download

Literature

This document is based on information from SAP Online Library, Implementation Guide of SAP Solution Manager 7.1, several SAP Notes and several SCN articles. These piece of information were enriched by the authors knowledge and experience.

Disclaimer

http://www.hybrid-eichhörnchen.de/rechtliche-hinweise/

Contact

Peter Weigel
Hyazinthenstr. 6
D-06122 Halle / Saale
Phone: +49 170 5337567
E-Mail: peter.weigel@hybrid-eichhoernchen.de
Web: www.hybrid-eichhoernchen.de

Download

You are allowed to download the latest version of this document for free: www.hybrid-eichhoernchen.de.

2 Configuration Documentation

2.1 Action Definition

We need to create PPF actions triggering the wanted method, for example a status change.

2.2 Schedule or Start Condition

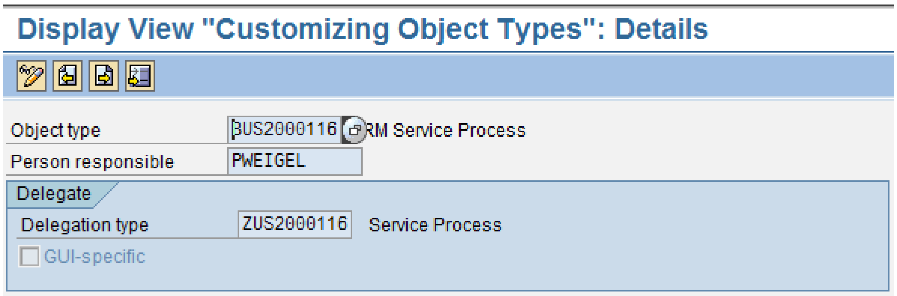
We need to schedule the previously created action. When we create schedule or start conditions we are able to evaluate all attributes of BOR object BUS2000116 (i.e. current user status or error free flag) including all attribute enhancements.

3 Development Documentation

3.1 Delegation

If there exist no enhancement yet:

- Call transaction SWO1.
- Create a new object ZUS2000116 “Service Process” as sub object of BUS2000116.
- Configure delegation from BUS2000116 to ZUS2000116 (SWO1 -> Settings -> Delegation).



Display View "Customizing Object Types": Details

Object type	BUS2000116 RM Service Process
Person responsible	PWEIGEL

Delegate

Delegation type	ZUS2000116 Service Process

☐ GUI-specific

If there exist an enhancement yet and it is allowed to adjust this enhancement:

- Lookup the current delegation.
- Use the existing Z BOR object.

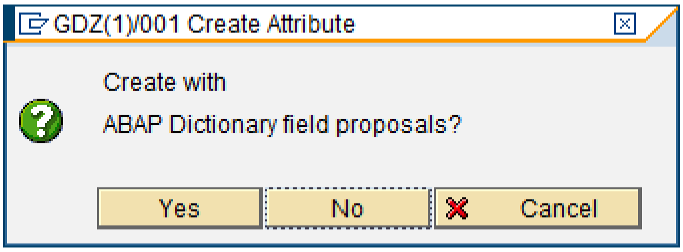
If there exist an enhancement yet and it is NOT allowed to adjust this enhancement:

- Call transaction SWO1.
- Create a new object ZUS2000116 “Service Process” as sub object of BUS2000116.
- Change subobject of the existing enhancement object. (hierarchy: BUS2000116 -> ZUS2000116 -> Object used for Delegation)

3.2 Simple Attribute Enhancement (AdminH, New)

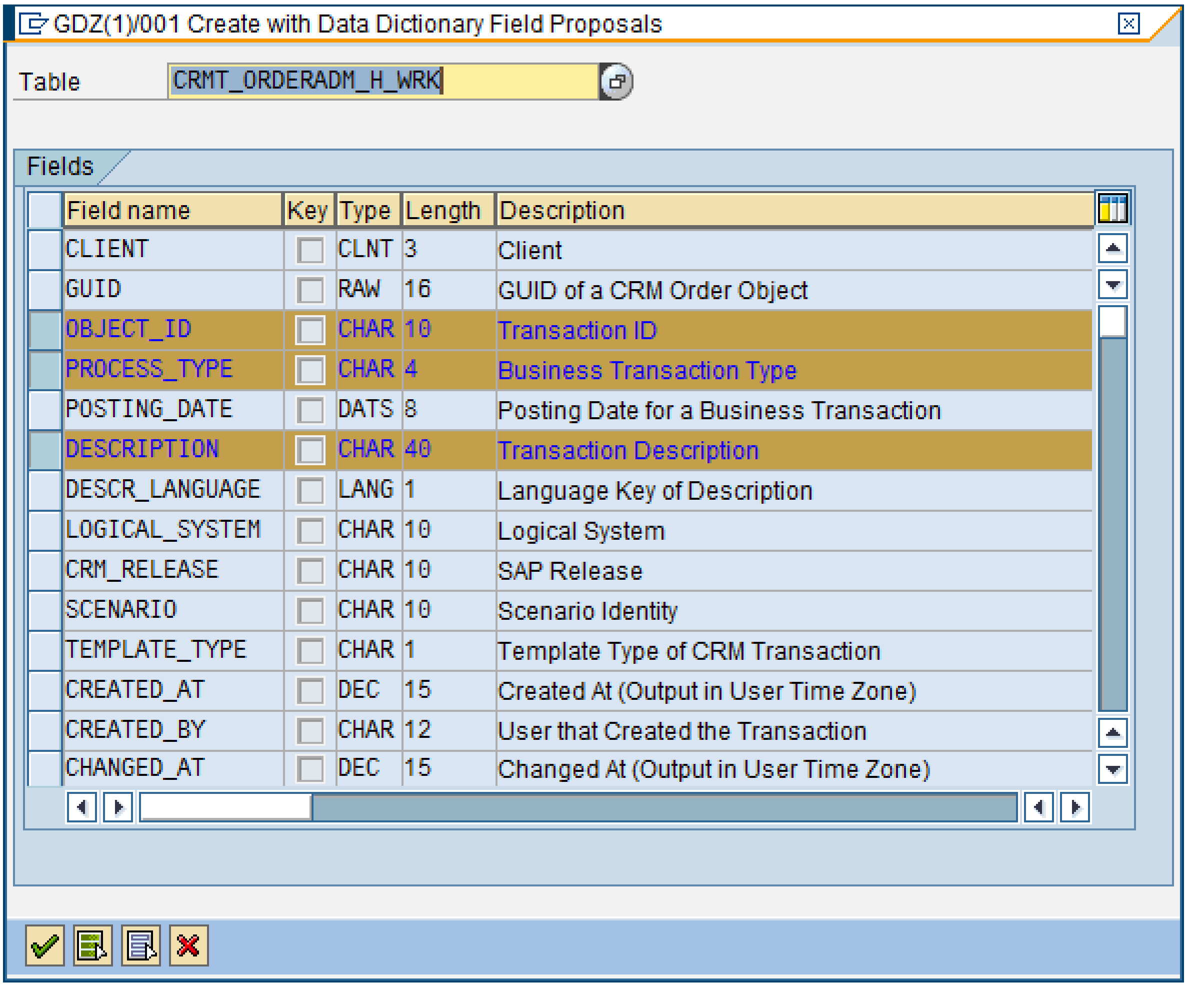
To get the current value of a field from ORDERADM_H, we always use structure CRMT_ORDERADM_H_WRK.

- In case we need an additional attribute of ORDERADM_H, we just need to create a new attribute as part of BOR object ZUS2000116.
- Here we confirm the question “Create with ABAP Dictionary proposal?” with “Yes”.
- We select structure CRMT_ORDERADM_H_WRK.
- We specify all needed information like attribute name.



GDZ(1)/001 Create Attribute

Create with
ABAP Dictionary field proposals?



GDZ(1)/001 Create with Data Dictionary Field Proposals

Table: CRMT_ORDERADM_H_WRK

Fields

Field name	Key	Type	Length	Description
CLIENT	☐	CLNT	3	Client
GUID	☐	RAW	16	GUID of a CRM Order Object
OBJECT_ID	☐	CHAR	10	Transaction ID
PROCESS_TYPE	☐	CHAR	4	Business Transaction Type
POSTING_DATE	☐	DATS	8	Posting Date for a Business Transaction
DESCRIPTION	☐	CHAR	40	Transaction Description
DESCR_LANGUAGE	☐	LANG	1	Language Key of Description
LOGICAL_SYSTEM	☐	CHAR	10	Logical System
CRM_RELEASE	☐	CHAR	10	SAP Release
SCENARIO	☐	CHAR	10	Scenario Identity
TEMPLATE_TYPE	☐	CHAR	1	Template Type of CRM Transaction
CREATED_AT	☐	DEC	15	Created At (Output in User Time Zone)
CREATED_BY	☐	CHAR	12	User that Created the Transaction
CHANGED_AT	☐	DEC	15	Changed At (Output in User Time Zone)

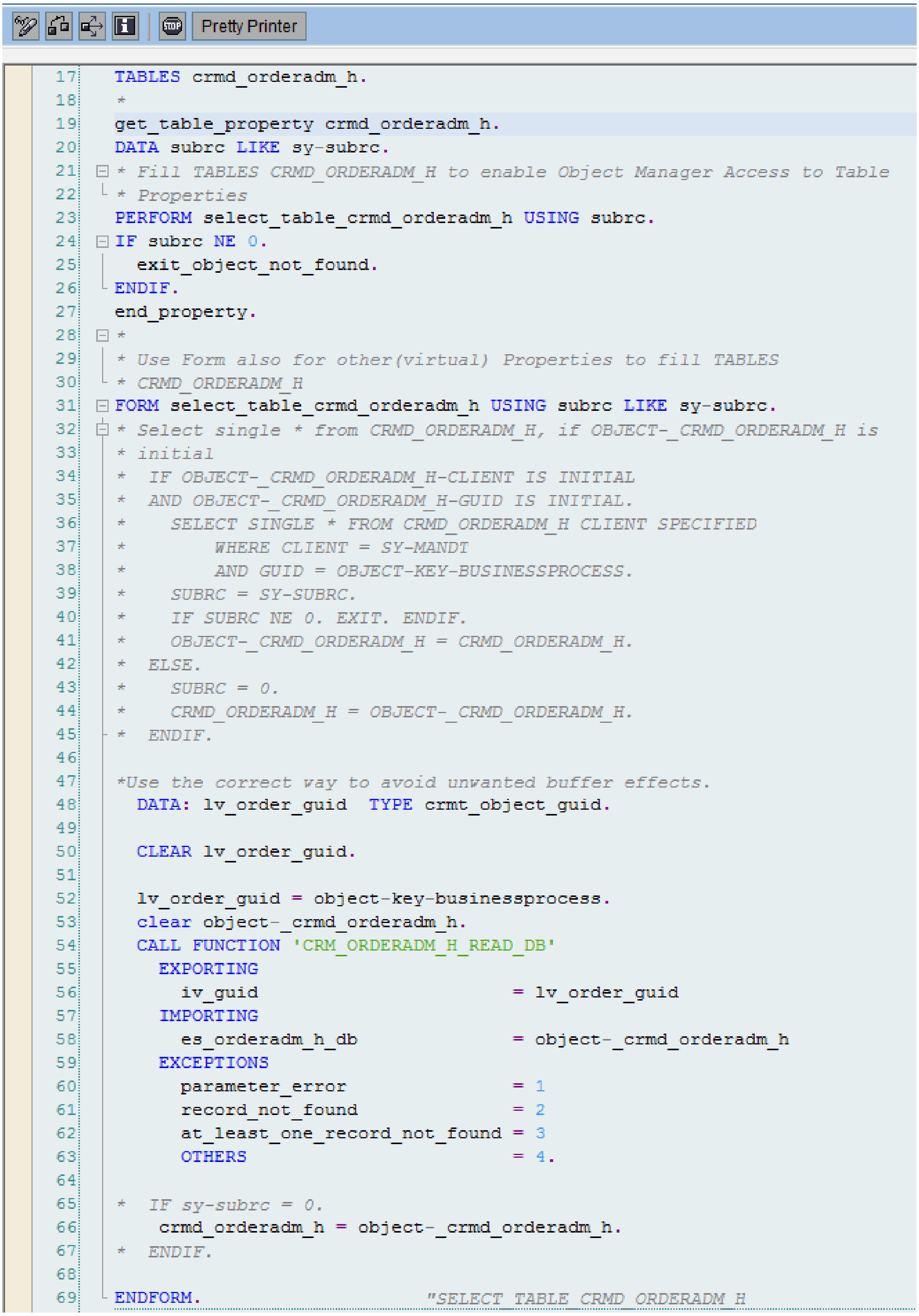
- After that we generate source code by going to program of this attribute.
- We change status of attribute and object to “implemented”.

- We generate object ZUS2000116.
- Now the new Attribute can be evaluated in schedule and start conditions.
- If more than one attributes are needed, we could repeat these steps.

3.3 Simple Attribute Enhancement (AdminH, Old)

To get the database value of a field from ORDERADM_H, we use structure CRMD_ORDERADM_H. The generated source code needs to be adjusted the first time this table/structure is used.

Object Type: Editor Display Program ZUS2000116



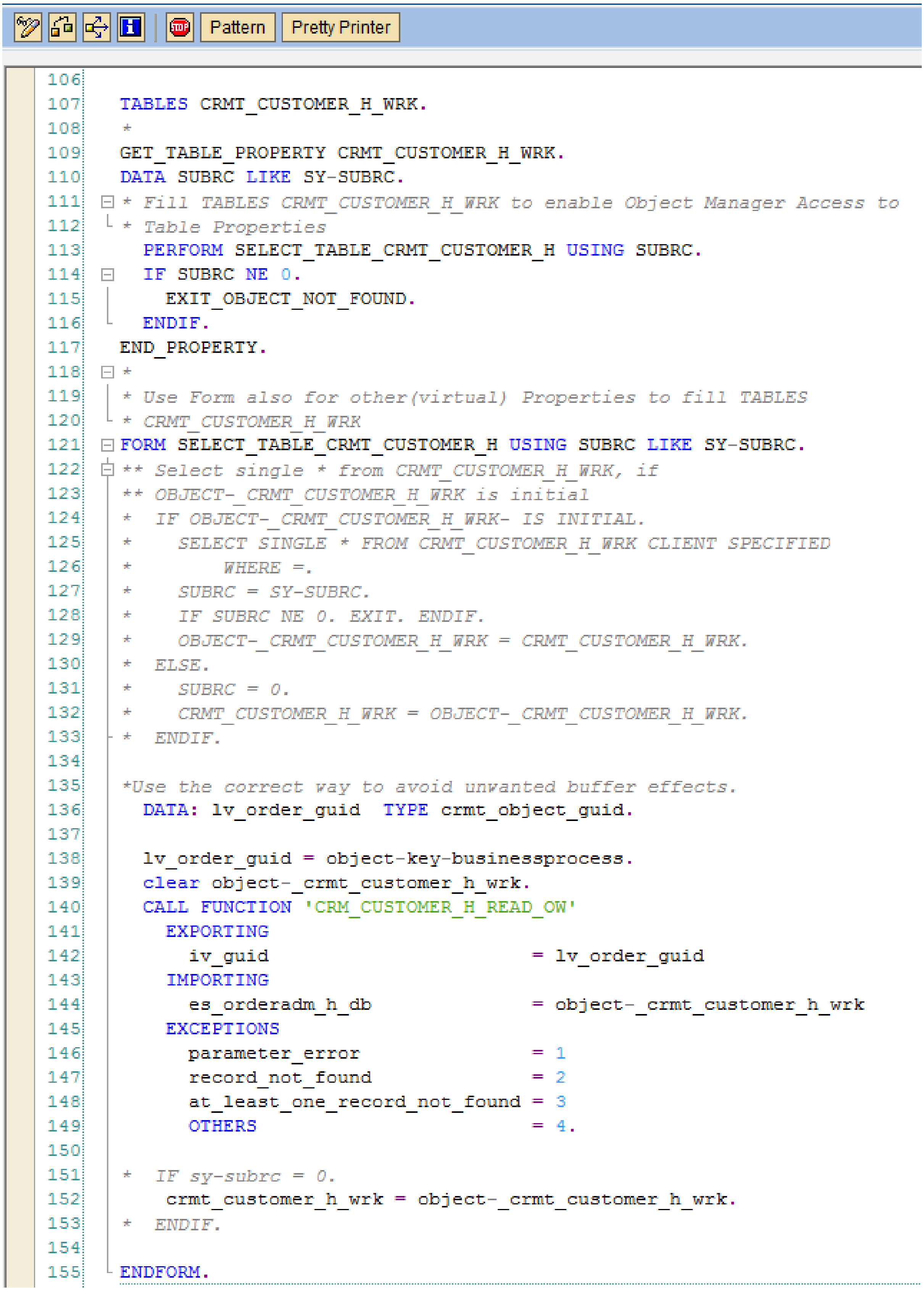
```
TABLES crmd_orderadm_h.
*
get_table_property crmd_orderadm_h.
DATA subrc LIKE sy-subrc.
* Fill TABLES CRMD_ORDERADM_H to enable Object Manager Access to Table
* Properties
PERFORM select_table_crmd_orderadm_h USING subrc.
IF subrc NE 0.
  exit_object_not_found.
ENDIF.
end_property.
*
* Use Form also for other(virtual) Properties to fill TABLES
* CRMD_ORDERADM_H
FORM select_table_crmd_orderadm_h USING subrc LIKE sy-subrc.
* Select single * from CRMD_ORDERADM_H, if OBJECT-_CRMD_ORDERADM_H is
* initial
*  IF OBJECT-_CRMD_ORDERADM_H-CLIENT IS INITIAL
*  AND OBJECT-_CRMD_ORDERADM_H-GUID IS INITIAL.
*    SELECT SINGLE * FROM CRMD_ORDERADM_H CLIENT SPECIFIED
*        WHERE CLIENT = SY-MANDT
*        AND GUID = OBJECT-KEY-BUSINESSPROCESS.
*    SUBRC = SY-SUBRC.
*    IF SUBRC NE 0. EXIT. ENDIF.
*    OBJECT-_CRMD_ORDERADM_H = CRMD_ORDERADM_H.
*  ELSE.
*    SUBRC = 0.
*    CRMD_ORDERADM_H = OBJECT-_CRMD_ORDERADM_H.
*  ENDIF.

*Use the correct way to avoid unwanted buffer effects.
  DATA: lv_order_guid  TYPE crmt_object_guid.

  CLEAR lv_order_guid.

  lv_order_guid = object-key-businessprocess.
  clear object-_crmd_orderadm_h.
  CALL FUNCTION 'CRM_ORDERADM_H_READ_DB'
    EXPORTING
      iv_guid                       = lv_order_guid
    IMPORTING
      es_orderadm_h_db              = object-_crmd_orderadm_h
    EXCEPTIONS
      parameter_error               = 1
      record_not_found              = 2
      at_least_one_record_not_found = 3
      OTHERS                        = 4.

*  IF sy-subrc = 0.
    crmd_orderadm_h = object-_crmd_orderadm_h.
*  ENDIF.

ENDFORM.                    "SELECT_TABLE_CRMD_ORDERADM_H
```

```
*Use the correct way to avoid unwanted buffer effects.
  DATA: lv_order_guid  TYPE crmt_object_guid.

  lv_order_guid = object-key-businessprocess.
  clear object-_crmd_orderadm_h.
  CALL FUNCTION 'CRM_ORDERADM_H_READ_DB'
    EXPORTING
      iv_guid                       = lv_order_guid
    IMPORTING
      es_orderadm_h_db              = object-_crmd_orderadm_h
    EXCEPTIONS
      parameter_error               = 1
      record_not_found              = 2
      at_least_one_record_not_found = 3
      OTHERS                        = 4.

*  IF sy-subrc = 0.
    crmd_orderadm_h = object-_crmd_orderadm_h.
*  ENDIF.
```

3.4 Simple Attribute Enhancement (CustomerH, New)

To get the current value of a field from CUSTOMER_H, we use structure CRMT_CUSTOMER_H_WRK (used by function module CRM_CUSTOMER_H_READ_OW). The generated source code needs to be adjusted the first time this table/structure is used.

Object Type: Editor Edit Program ZUS2000116

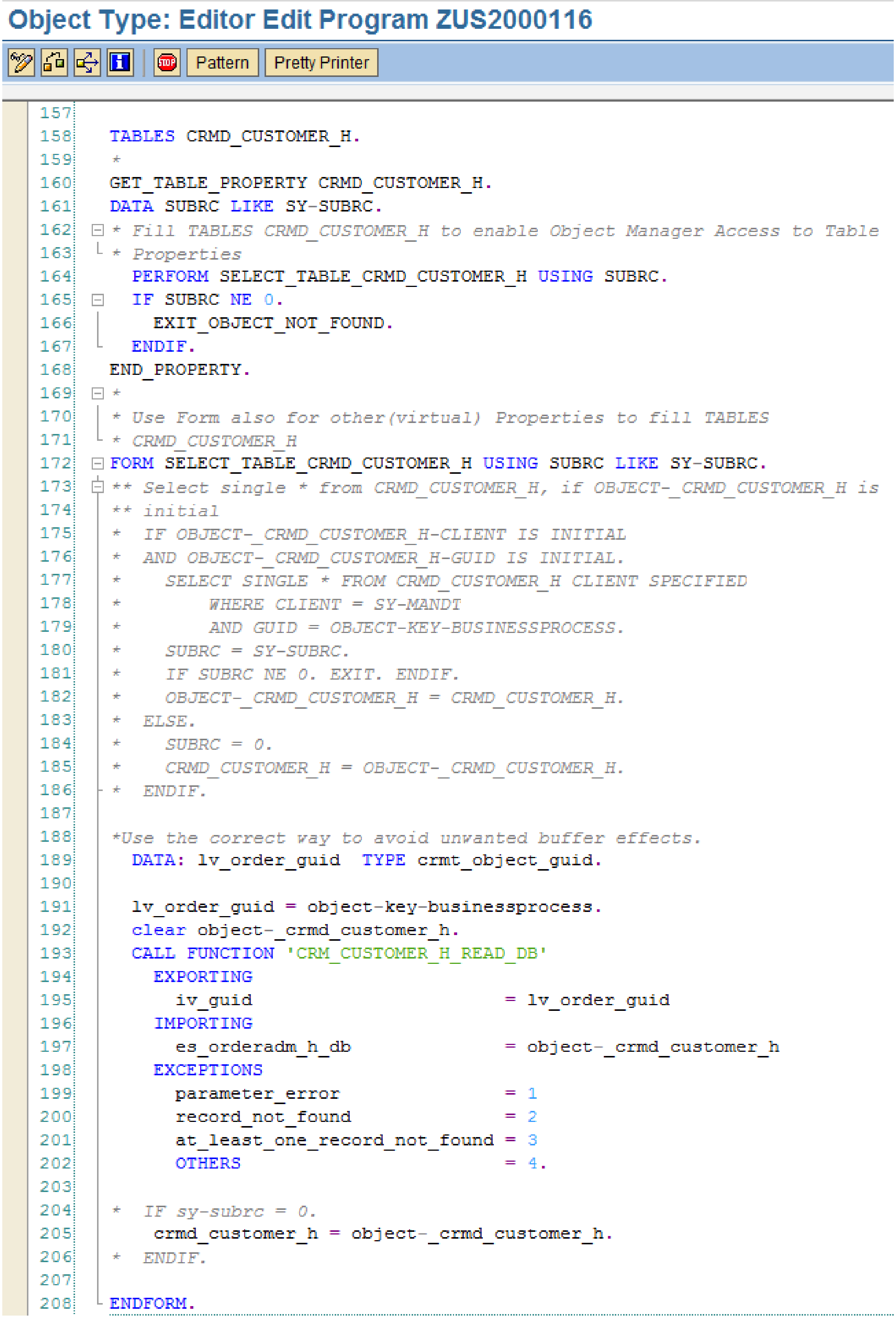


```

TABLES CRMT_CUSTOMER_H_WRK.
*
GET_TABLE_PROPERTY CRMT_CUSTOMER_H_WRK.
DATA SUBRC LIKE SY-SUBRC.
* Fill TABLES CRMT_CUSTOMER_H_WRK to enable Object Manager Access to
* Table Properties
  PERFORM SELECT_TABLE_CRMT_CUSTOMER_H USING SUBRC.
  IF SUBRC NE 0.
    EXIT_OBJECT_NOT_FOUND.
  ENDIF.
END_PROPERTY.
*
* Use Form also for other(virtual) Properties to fill TABLES
* CRMT_CUSTOMER_H_WRK
FORM SELECT_TABLE_CRMT_CUSTOMER_H USING SUBRC LIKE SY-SUBRC.
** Select single * from CRMT_CUSTOMER_H_WRK, if
** OBJECT-_CRMT_CUSTOMER_H_WRK is initial
*  IF OBJECT-_CRMT_CUSTOMER_H_WRK- IS INITIAL.
*    SELECT SINGLE * FROM CRMT_CUSTOMER_H_WRK CLIENT SPECIFIED
*        WHERE =.
*    SUBRC = SY-SUBRC.
*    IF SUBRC NE 0. EXIT. ENDIF.
*    OBJECT-_CRMT_CUSTOMER_H_WRK = CRMT_CUSTOMER_H_WRK.
*  ELSE.
*    SUBRC = 0.
*    CRMT_CUSTOMER_H_WRK = OBJECT-_CRMT_CUSTOMER_H_WRK.
*  ENDIF.

*Use the correct way to avoid unwanted buffer effects.
  DATA: lv_order_guid  TYPE crmt_object_guid.

  lv_order_guid = object-key-businessprocess.
  clear object-_crmt_customer_h_wrk.
  CALL FUNCTION 'CRM_CUSTOMER_H_READ_OW'
    EXPORTING
      iv_guid                       = lv_order_guid
    IMPORTING
      es_orderadm_h_db              = object-_crmt_customer_h_wrk
    EXCEPTIONS
      parameter_error               = 1
      record_not_found              = 2
      at_least_one_record_not_found = 3
      OTHERS                        = 4.

*  IF sy-subrc = 0.
    crmt_customer_h_wrk = object-_crmt_customer_h_wrk.
*  ENDIF.

ENDFORM.
```

```
*Use the correct way to avoid unwanted buffer effects.
  DATA: lv_order_guid  TYPE crmt_object_guid.

  lv_order_guid = object-key-businessprocess.
  clear object-_crmt_customer_h_wrk.
  CALL FUNCTION 'CRM_CUSTOMER_H_READ_OW'
    EXPORTING
      iv_guid                       = lv_order_guid
    IMPORTING
      ES_CUSTOMER_H_WRK             = object-_crmt_customer_h_wrk
    EXCEPTIONS
      parameter_error               = 1
      record_not_found              = 2
      at_least_one_record_not_found = 3
      OTHERS                        = 4.

*  IF sy-subrc = 0.
    crmt_customer_h_wrk = object-_crmt_customer_h_wrk.
*  ENDIF.
```

3.5 Simple Attribute Enhancement (CustomerH, Old)

To get the database value of a field from CUSTOMER_H, we use structure CRMD_CUSTOMER_H (used by function module CRM_CUSTOMER_H_READ_DB). The generated source code needs to be adjusted the first time this table/structure is used.



```

TABLES CRMD_CUSTOMER_H.
*
GET_TABLE_PROPERTY CRMD_CUSTOMER_H.
DATA SUBRC LIKE SY-SUBRC.
* Fill TABLES CRMD_CUSTOMER_H to enable Object Manager Access to Table
* Properties
  PERFORM SELECT_TABLE_CRMD_CUSTOMER_H USING SUBRC.
  IF SUBRC NE 0.
    EXIT_OBJECT_NOT_FOUND.
  ENDIF.
END_PROPERTY.
*
* Use Form also for other(virtual) Properties to fill TABLES
* CRMD_CUSTOMER_H
FORM SELECT_TABLE_CRMD_CUSTOMER_H USING SUBRC LIKE SY-SUBRC.
** Select single * from CRMD_CUSTOMER_H, if OBJECT-_CRMD_CUSTOMER_H is
** initial
*  IF OBJECT-_CRMD_CUSTOMER_H-CLIENT IS INITIAL
*  AND OBJECT-_CRMD_CUSTOMER_H-GUID IS INITIAL.
*    SELECT SINGLE * FROM CRMD_CUSTOMER_H CLIENT SPECIFIED
*        WHERE CLIENT = SY-MANDT
*        AND GUID = OBJECT-KEY-BUSINESSPROCESS.
*    SUBRC = SY-SUBRC.
*    IF SUBRC NE 0. EXIT. ENDIF.
*    OBJECT-_CRMD_CUSTOMER_H = CRMD_CUSTOMER_H.
*  ELSE.
*    SUBRC = 0.
*    CRMD_CUSTOMER_H = OBJECT-_CRMD_CUSTOMER_H.
*  ENDIF.

*Use the correct way to avoid unwanted buffer effects.
  DATA: lv_order_guid  TYPE crmt_object_guid.

  lv_order_guid = object-key-businessprocess.
  clear object-_crmd_customer_h.
  CALL FUNCTION 'CRM_CUSTOMER_H_READ_DB'
    EXPORTING
      iv_guid                       = lv_order_guid
    IMPORTING
      es_orderadm_h_db              = object-_crmd_customer_h
    EXCEPTIONS
      parameter_error               = 1
      record_not_found              = 2
      at_least_one_record_not_found = 3
      OTHERS                        = 4.

*  IF sy-subrc = 0.
    crmd_customer_h = object-_crmd_customer_h.
*  ENDIF.

ENDFORM.
```

```
*Use the correct way to avoid unwanted buffer effects.
  DATA: lv_order_guid  TYPE crmt_object_guid.

  lv_order_guid = object-key-businessprocess.
  clear object-_crmd_customer_h.
  CALL FUNCTION 'CRM_CUSTOMER_H_READ_DB'
    EXPORTING
      iv_guid                       = lv_order_guid
    IMPORTING
      ES_CUSTOMER_H_DB              = object-_crmd_customer_h
    EXCEPTIONS
      parameter_error               = 1
      record_not_found              = 2
      at_least_one_record_not_found = 3
      OTHERS                        = 4.

*  IF sy-subrc = 0.
    crmd_customer_h = object-_crmd_customer_h.
*  ENDIF.
```

3.6 Maintained Texts

Sometimes we want to trigger an action if a specific text type is maintained. If the text type is configured as type “P”, the text will be added to text log on save. The maintained text check will work in this case only before or on save. After save the text type is empty again.

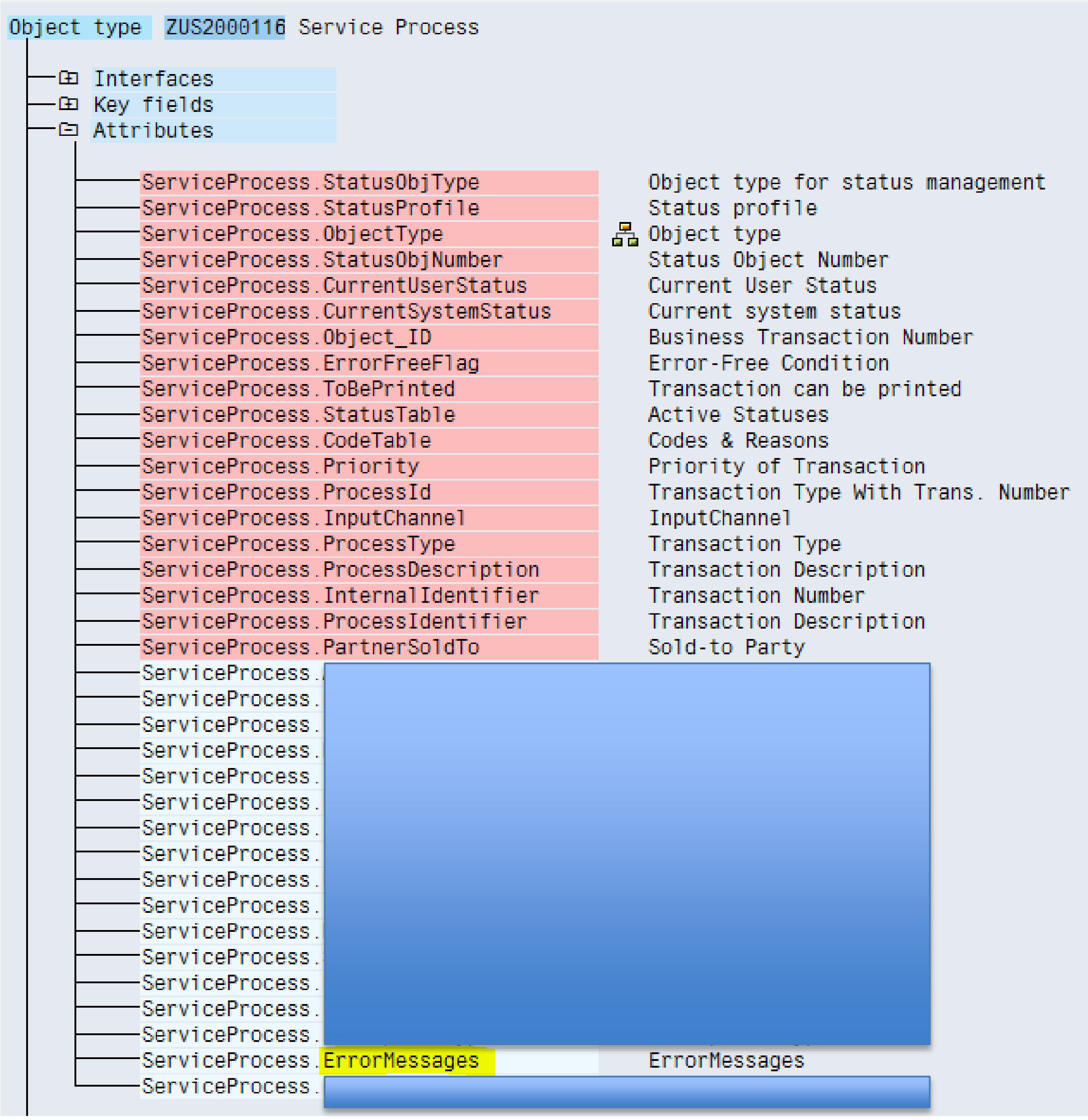
```
get_property maintainedtexts changing container.

DATA:
  lv_guid       TYPE crmt_object_guid,
  lt_guid       TYPE crmt_object_guid_tab,
  lt_text       TYPE crmt_text_wrkt.

FIELD-SYMBOLS:
  <fs_text> LIKE LINE OF lt_text[].

*  Ensure initial result.
CLEAR:
  object-maintainedtexts[],
  lt_guid[],
  lt_text[].

*  Get maintained texts.
lv_guid = object-key-businessprocess.
APPEND lv_guid TO lt_guid[].
CALL FUNCTION 'CRM_TEXT_READ_API'
  EXPORTING
    it_guid             = lt_guid[]
    iv_object_kind      = 'A'
*   IV_BUILD_INT_TABLES = FALSE
*   IV_NO_AUTH_CHECK    = FALSE
  IMPORTING
    et_text             = lt_text[].
```

```
*Extract text types.
LOOP AT lt_text[] ASSIGNING <fs_text>.
  APPEND <fs_text>-stxh-tdid TO object-maintainedtexts[].
ENDLOOP.

*Return result.
swc_set_table container 'MaintainedTexts' object-maintainedtexts.
end_property.
```

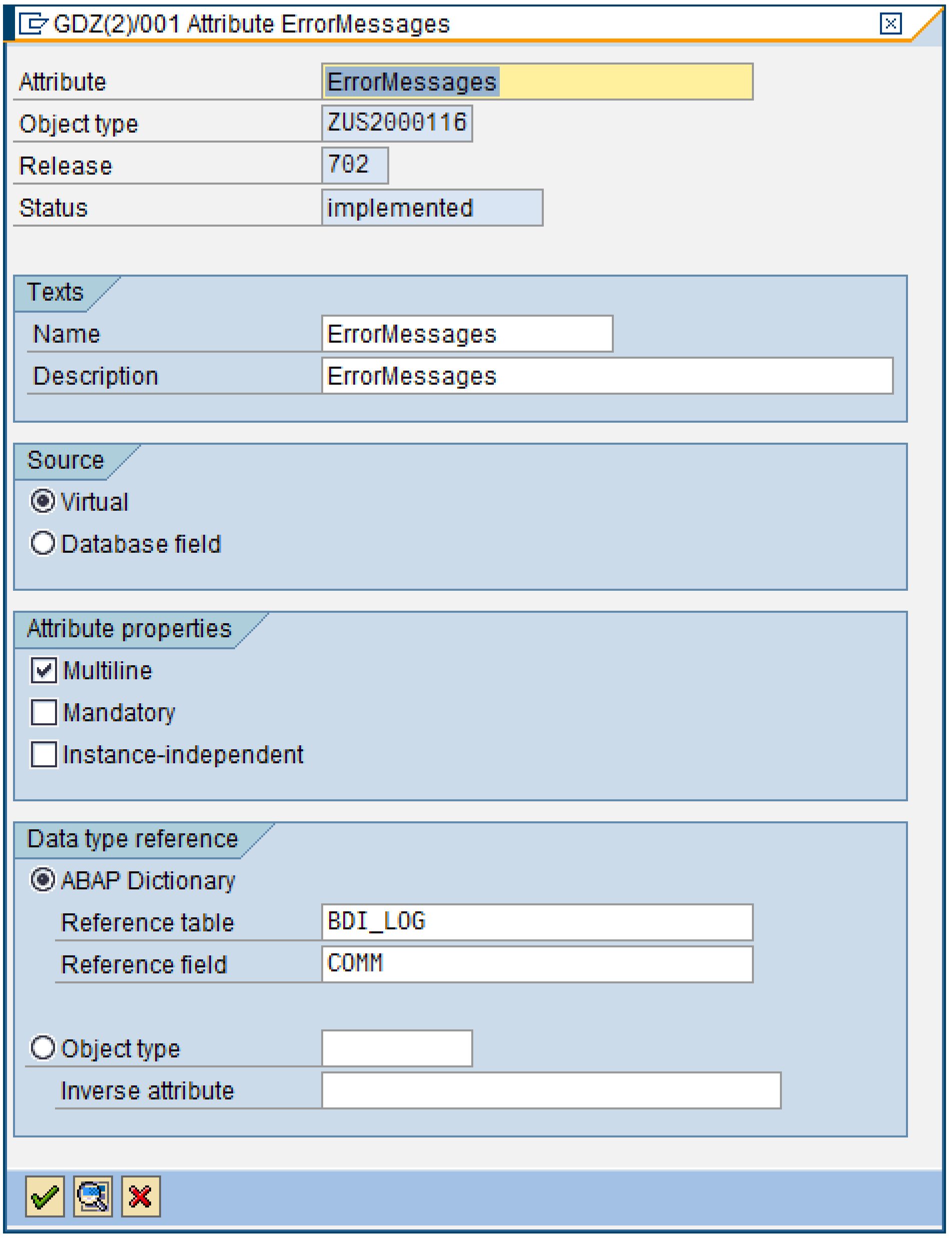
3.7 Error Messages

It is now possible to check for error messages in schedule or start condition for PPF actions. We can therefore send e-mail notifications as soon as specific error messages occur in business transaction application log.

There exist a new multi-line transaction attribute "ErrorMessages" which contains all error messages of the specific business transaction in format "/MSGTY:<TYPE>/MSGID:<ID>/MSGNO:<NO>". It is possible to check for complete entries as well as to use wildcards "+" and "*" to check for specific parts only.



```
Object type  ZUS2000116 Service Process

  ⊞ Interfaces
  ⊞ Key fields
  ⊟ Attributes
      ServiceProcess.StatusObjType          Object type for status management
      ServiceProcess.StatusProfile          Status profile
      ServiceProcess.ObjectType             Object type
      ServiceProcess.StatusObjNumber        Status Object Number
      ServiceProcess.CurrentUserStatus      Current User Status
      ServiceProcess.CurrentSystemStatus    Current system status
      ServiceProcess.Object_ID              Business Transaction Number
      ServiceProcess.ErrorFreeFlag          Error-Free Condition
      ServiceProcess.ToBePrinted            Transaction can be printed
      ServiceProcess.StatusTable            Active Statuses
      ServiceProcess.CodeTable              Codes & Reasons
      ServiceProcess.Priority               Priority of Transaction
      ServiceProcess.ProcessId              Transaction Type With Trans. Number
      ServiceProcess.InputChannel           InputChannel
      ServiceProcess.ProcessType            Transaction Type
      ServiceProcess.ProcessDescription     Transaction Description
      ServiceProcess.InternalIdentifier     Transaction Number
      ServiceProcess.ProcessIdentifier      Transaction Description
      ServiceProcess.PartnerSoldTo          Sold-to Party
      ServiceProcess.[redacted]
      ServiceProcess.
      ServiceProcess.
      ServiceProcess.
      ServiceProcess.
      ServiceProcess.
      ServiceProcess.
      ServiceProcess.
      ServiceProcess.
      ServiceProcess.
      ServiceProcess.
      ServiceProcess.
      ServiceProcess.
      ServiceProcess.
      ServiceProcess.
      ServiceProcess.ErrorMessages          ErrorMessages
      ServiceProcess.[redacted]
```



GDZ(2)/001 Attribute ErrorMessages

Attribute: ErrorMessages
Object type: ZUS2000116
Release: 702
Status: implemented

Texts
Name: ErrorMessages
Description: ErrorMessages

Source
● Virtual
○ Database field

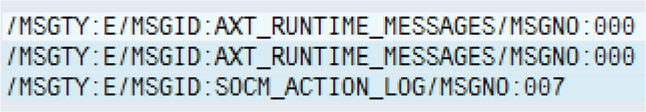
Attribute properties
☑ Multiline
□ Mandatory
□ Instance-independent

Data type reference
● ABAP Dictionary
Reference table: BDI_LOG
Reference field: COMM
○ Object type:
Inverse attribute:

```
get_property errormessages changing container.

DATA:
  lv_guid          TYPE crmt_object_guid,
  lt_msg_handle    TYPE bal_t_msgh,
  ls_msg_info      TYPE crmt_msg_info,
  ls_msg           TYPE bal_s_msg,
  lv_errormessage LIKE LINE OF object-errormessages.

FIELD-SYMBOLS:
  <fs_msg_handle> LIKE LINE OF lt_msg_handle.

CLEAR object-errormessages.

*  Extract GUID of Transaction
lv_guid = object-key-businessprocess.

*  Get Error Messages.
```

```
CALL FUNCTION 'CRM_MESSAGES_SEARCH'
  EXPORTING
*   it_r_msgidno     = lt_msg_idno[]
    iv_ref_object    = lv_guid
    iv_ref_kind      = 'A'
*   IV_CALLER_NAME   =
*   IT_LOGICAL_KEYS  =
*   IV_PROBCLASS     =
*   IV_DETLEVEL      =
  IMPORTING
    et_msg_handle    = lt_msg_handle
  EXCEPTIONS
    appl_log_error   = 1
    error_occurred   = 2
    OTHERS           = 3.

*  We found some messages.
IF sy-subrc = 0 AND lt_msg_handle[] IS NOT INITIAL.

*  Process every single message.
  LOOP AT lt_msg_handle ASSIGNING <fs_msg_handle>.

*  Get message details.
    CALL FUNCTION 'CRM_MESSAGES_GET_MSG_INFO'
      EXPORTING
        is_msg_handle            = <fs_msg_handle>
*       IV_GET_CALLER_NAME       = TRUE
      IMPORTING
        es_info                  = ls_msg_info
        es_msg                   = ls_msg
      EXCEPTIONS
        not_found                = 1
        wrong_context_structure = 2
        data_error               = 3
        OTHERS                   = 4.

    IF sy-subrc = 0.

*  Build result line.
      lv_errormessage = '/MSGTY:' && ls_msg-msgty &&
                        '/MSGID:' && ls_msg-msgid &&
                        '/MSGNO:' && ls_msg-msgno.

      APPEND lv_errormessage TO object-errormessages.

    ENDIF.

  ENDLOOP.

ENDIF.

swc_set_table container 'ErrorMessages' object-errormessages.

end_property.
```

Result Example:



```
/MSGTY:E/MSGID:AXT_RUNTIME_MESSAGES/MSGNO:000
/MSGTY:E/MSGID:AXT_RUNTIME_MESSAGES/MSGNO:000
/MSGTY:E/MSGID:SOCM_ACTION_LOG/MSGNO:007
```

4 Alternative Solutions

4.1 BADI for Schedule or Start Conditions

If you want to check complex conditions which you don´t want to combine with simple attribute checks, you can implement a BADI implementation for Schedule (EVAL_SCHEDCOND_PPF) or Start Conditions (EVAL_STARTCOND_PPF). At action definitions you have to switch to BADI Conditions instead of Workflow Conditions. At action scheduling you have to select the previously implemented BADI implementation.

Please note that this solution is working, but you are not able to combine these checks with simple conditions like “error free” or “user status”.

4.2 BADI for PPF Container Parameter

If you want to provide attributes resp. parameters which should not be visible to anyone, you can use PPF container attributes. At action scheduling you have to define and to use a parameter. In implementation of BADI CONTAINER_PPF you can now check whether this parameter is requested by a schedule or start condition. If yes, you can now read and calculate the value for it.

Please note, that this solution is working fine, but you need to know which parameters are existing, because you need to define it in every schedule or start condition you want to use it.

Example: http://www.hybrid-eichhoernchen.de/checking-landscape-information-using-badi-container_ppf/