Applies to

Master Data Governance for Financials (MDG-F) with release versions 7.0 and newer. For more information, visit the Master Data Management homepage (https://go.sap.com/community/topic/master-data-governance.html).

Summary

SAP Master Data Governance provides out-of-the-box solutions for the central management of master data objects. Domain-specific solutions include business partner (MDG-BP), customer (MDG-C), supplier (MDG-S) governance, material governance (MDG-M), and financials governance (MDG-F).

This guide provides you with the foundation knowledge you need to know about financial data and its related governance solution financial governance (MDG-F).

Author(s): Michael Theis
Company: SAP SE
Created on: May 2017
Version: 1.2
Table of Contents

Applies to ................................................................................................................................. 1

Summary ..................................................................................................................................... 1

Introduction ............................................................................................................................... 3

  MDG for Financials .................................................................................................................... 3

  Scenario .................................................................................................................................. 3

Steps for the ALE Scenario Configuration ................................................................................... 4

  Define Logical Systems .......................................................................................................... 4

  Define a RFC Connection ....................................................................................................... 5

  Define an ALE tRFC Port ....................................................................................................... 5

  Define an ALE Distribution Model ........................................................................................ 6

  Define Partner Profiles ......................................................................................................... 7

Steps for the MDG Data Replication Framework (DRF) Configuration ....................................... 10

  Define Business Systems ..................................................................................................... 10

  Define a Replication Model .................................................................................................. 11

Confirmation IDocs (ALEAUD) for setting replication status in MDG ..................................... 12

Additional Information ............................................................................................................ 13

  Links .................................................................................................................................... 13

  How-to Guides ....................................................................................................................... 13

  SAP Notes ............................................................................................................................. 13

  Version History .................................................................................................................... 13

Appendix ................................................................................................................................. 14

Copyright ..................................................................................................................................... 33
Introduction

SAP Master Data Governance (MDG) is used for embedded Master Data Management (MDM), that is, centralized, out-of-the-box, domain-specific creation, modification, and distribution of master data with a focus on SAP Business Suite.

For the replication of master data created or changed in MDG into the client systems (ERP) SAP recommends the usage of enterprise services or ALE IDocs. The standard replication scenarios require that MDG hub and MDG client are two different systems, or at least two different physical clients within the same system.

For MDG-F, however, customers might want to run the MDG hub in the same physical client as the ERP client due to the nature of the MDG-F data model. Since all MDG-F entity types use the flexible mode their data is stored only in the MDG generated tables and not within ERP tables.

This document explains how-to setup an ALE scenario for data replication where MDG hub and ERP client share the same physical client of the same system.

We recommend that you study the following how-to guide before working with the current guide:

Extensibility Options for SAP Master Data Governance for Financial Data ➔ Overview about MDG-F

Strong ALE configuration expertise is required, too.

MDG for Financials

MDG offers a domain specific solution for financial governance (MDG-F). The current MDG-F data model is called G. It covers entity types of the accounting, controlling and consolidation components of financial master data as indicated by the examples below:

- Accounting: G/L Account (**ACCOUNT & ACCCCDET**), Company (**COMPANY**)
- Controlling: Cost Center (**CCTR**), Cost Element (**CELEM**) and Profit Center (**PCTR**)
- Consolidation: Consolidation Unit (**CONSUNIT**), Item (**FSI**)

Scenario

The replication into the same client requires setting up both ALE and MDG Data Replication Framework. The configuration is described in the next chapters.
Steps for the ALE Scenario Configuration

The guide uses the system E8B and its client 001 as sample data. When you configure this scenario for your landscape, ensure you replace the system ID and the client ID with your own system data.

Define Logical Systems

Start transaction SALE. In the tree menu navigate to Basic Settings -> Logical Systems -> Define Logical System.

Define two logical systems:

- One for the MDG hub (in this example E8BCLNT001)
- One for the MDG client (in this example E8B_001)
Define a RFC Connection

Start transaction SALE. In the tree menu navigate to Communication -> Create RFC Connections (alternatively start transaction SM59).

Create a RFC connection using Connection Type 3 (ABAP Connection) into the same client.

Define an ALE tRFC Port

Start transaction WE21. Create a port for transactional RFC. The port uses the RFC destination created in the step before.
Define an ALE Distribution Model

Start transaction SALE. In the tree menu navigate to Modelling and Implementing Business Processes -> Maintain Distribution Model and Distribute Views (alternatively start transaction BD64).

Use the previously created logical systems as sender (MDG Hub) and receiver (MDG client). Add the required message types (e.g. COSMAS for cost centers) according to your replication scenario.
Define Partner Profiles

There are two options for the definition of partner profiles. If you are still in the Change Distribution View of the previous step, use the top menu bar and execute Environment -> Generate Partner Profiles. The screenshots below show needed parameters. Alternatively start transaction WE20.

Locate the MDG Client (in this example EBB_001) below the tree node Partner Type LS (Logical System) to maintain the settings for the IDoc receiver. Add all message types that you use in your replication scenario as Outbound Parameters.

![Partner profiles diagram]

- Partner Type B
- Partner Type BP
- Partner Type GP
- Partner Type KU
- Partner Type LI
- Partner Type LS
- EBBCLNT001
- EBBCLNT201
- EBB_001
- Partner Type LS

- Partner Role
- Message Type
- Message No.
- Message Function
- Test

- Partner Role
- Message Type
- Message No.
- Message Function
- Test
Define the previously created ALE tRFC Port for each message. This enables the connection to the own client as IDoc receiver.
Locate the MDG Hub (in this example E8BCLNT001) below the tree node **Partner Type LS (Logical System)** to maintain the settings for the IDoc sender. Add all message types that you use in your replication scenario as **Inbound Parameters**.
Steps for the MDG Data Replication Framework (DRF) Configuration

The customizing transactions for the steps in this section are collected in the IMG of the data replication framework. Use transaction DRFIMG to display the DRF implementation guide.

Define Business Systems

Navigate to Define Custom Settings for Data Replication -> Define Technical Settings -> Define Technical Settings for Business Systems.

Create a Business System for the IDoc Receiver (in this example Logical System E8B_001). Use the RFC Destination to the own client.

Assign the related Business Object Types for all MDG-F entity types you want to replicate (for example, 158 Cost Center).
Specify the communication channel for IDoc replication for each Business Object Type.

**Define a Replication Model**

Navigate to Define Custom Settings for Data Replication -> Define Replication Models.

Assign your own client (in this example E8B_001) as the Target Business System for each IDoc Replication Model and Outbound Implementation.

Activate the replication model.
Confirmation IDocs (ALEAUD) for setting replication status in MDG

In standard ALE replication scenarios where sender and receiving system are at least located in different physical clients you can use IDocs of message type ALEAUD for setting the replication status in MDG DRF. These ALE audit IDocs are triggered by report RBDSTATE which periodically runs in a background job.

In this special scenario where you have only one physical client this would require an ALE replication model for IDOC type ALEAUD with sender system = receiver system. Unfortunately this is not allowed in transaction BD64.

As a workaround you can create a copy of program RBDSTATE which sends ALE audit documents within the same client without having the distribution model for ALEAUD defined in BD64. MDG specific enhancements are required in the copied report. Check the source code in the appendix.
Additional Information

Links

MDG Guides on Service Market Place

How-to Guides

Extensibility Options for SAP Master Data Governance for Financial Data

SAP Notes

- [1637249](#) specifying required information for OSS support
- [2105467](#) specifying required information for Performance Issues

Version History

- 1.1 – New MDG 9.0 related Information


Appendix

REPORT zmdgrbdstate MESSAGE-ID b1.
** Copy of program RBDSTATE - Use for MDG only!
** Required for sending ALEAUDIT if ALE distribution for MDG objects
** is setup within same physical client.
** MDG specific enhancements are marked with "MDG Enhancement"
**
** TODO:
** Copy GUI Status "STANDARD" from program RBDSTATE to this report!

INCLUDE mbdconst.
INCLUDE bdcstaud.

CONSTANTS:
  c_max_idocs TYPE i VALUE '500'.

*** Begin MDG Enhancement

TYPES:
  tyt_idoc_list TYPE TABLE OF bdidocs,
  tyt_idoc_contrl TYPE TABLE OF edidc,
  tyt_idoc_data TYPE TABLE OF edidd,
  tyt_idoc_status TYPE TABLE OF bdidocstat,
  tyt_edi_status_records TYPE TABLE OF edi_ds40.

DATA:
  gt_idoc_contrl TYPE tyt_idoc_contrl,
  gt_idoc_data TYPE tyt_idoc_data,
  gt_idoc_status TYPE tyt_idoc_status.
*** End MDG Enhancement

TABLES: edidc, sscrfields, bdaudstate.

DATA: gs_layout TYPE slis_layout_alv,
  gt_fieldcat TYPE slis_t_fieldcat_alv,
  gs_excluding TYPE slis_t_extab,
  gs_excl_head TYPE slis_extab,
g_status_set  TYPE slis_formname VALUE 'PF_STATUS_SET',
g_user_command TYPE slis_formname VALUE 'USER_COMMAND',
header TYPE lvc_title.

SELECT-OPTIONS:
  s_sndsys FOR bdaudstate-rcv_system,
  s_recsys FOR bdaudstate-snd_system.  "MDG Enhancement
SELECTION-SCREEN SKIP 1.
SELECT-OPTIONS:
  s_mestyp FOR bdaudstate-mess_type,
  s_mescod FOR bdaudstate-mess_code,
  s_mesfct FOR bdaudstate-mess_funct,
  status FOR edidc-status.
SELECTION-SCREEN SKIP 1.
SELECT-OPTIONS: s_upddat FOR edidc-upddat NO-EXTENSION 

DATA:
  left_date LIKE edidc-upddat,
  left_time LIKE edidc-updtim,
  right_date LIKE edidc-upddat,
  right_time LIKE edidc-updtim,
  t_idoc_control TYPE audit_idoc_control_tab,
  t_idoc_control_all TYPE audit_idoc_control_tab,
  s_idoc_control TYPE audit_idoc_control_t ,
  nothing_to_do,
  resulting_idocs LIKE bdidocs OCCURS 0 WITH HEADER LINE.

DATA: paket TYPE i,
  t_idoc_control_all_max TYPE i,
  t_idoc_control_all_anz TYPE i.

START-OF-SELECTION.

IF NOT sy-batch IS INITIAL AND s_upddat[] IS INITIAL.
  PERFORM time_interval_get
  CHANGING
    left_date
left_time
right_date
right_time.

MESSAGE i147 WITH left_date left_time right_date right_time.

* Selektionszeitraum für die AUDIT Idocs: &1 &2 -> &3 &4 .
* comparisons are only based on numbers (date and time). This is
* portable to AS/400.

SELECT sndprn mestyp mescod mesfct credat cretim docnum status
FROM edidc INTO TABLE t_idoc_control_all
WHERE ( upddat = left_date AND updtim >= left_time  "#EC PORTABLE
OR upddat > left_date ) "#EC PORTABLE
AND ( upddat < right_date                           "#EC PORTABLE
OR upddat = right_date AND updtim <= right_time )   "#EC PORTABLE
AND sndprt = c_prt Logical_system
AND sndprn IN s_sndsys
AND rcvprn IN s_recsys "MDG Enhancement
AND mestyp IN s_mestyp
AND mescod IN s_mescod
AND mesfct IN s_mesfct
AND status IN status
AND status <> c_status in archive reload
AND status <> c_status in archived
AND status <> c_status in orig_of edited
ORDER BY sndprn mestyp mesfct mescod credat cretim.

IF sy-subrc <> 0.
nothing_to_do = 'X'.
ELSE.

DESCRIBE TABLE t_idoc_control_all LINES t_idoc_control_all_max.

LOOP AT t_idoc_control_all INTO s_idoc_control.

APPEND s_idoc_control TO t_idoc_control.
t_idoc_control_all_anz = t_idoc_control_all_anz + 1.
paket = paket + 1.
IF paket = c_max_idocs OR
    t_idoc_control_all_anz = t_idoc_control_all_max.

PERFORM idocs_create
    TABLES
        resulting_idocs
    USING
        t_idoc_control.

CLEAR paket.
REFRESH t_idoc_control.
ENDIF.
ENDLOOP.

ENDIF.
ELSE.

MESSAGE s147 WITH
    s_upddat-low '00:00:00' s_upddat-high '24:00:00' .
* Selektionszeitraum für die AUDIT Idocs: &1 &2 -> &3 &4 .

SELECT sndprn mestyp mescod mesfct credat cretim docnum status
FROM edidc INTO TABLE t_idoc_control_all
WHERE upddat IN s_upddat
    AND sndprt = c_prt_logical_system
    AND sndprn IN s_sndsys
    AND rcvprn IN s_recsys
    "MDG Enhancement"
    AND mestyp IN s_mestyp
    AND mescod IN s_mescod
    AND mesfct IN s_mesfct
    AND status IN status
    AND status <> c_status_in_archive_reload
    AND status <> c_status_in_archived
    AND status <> c_status_in_orig_of_edited
ORDER BY sndprn mestyp mesfct mescod credat cretim.

IF sy-subrc <> 0.
nothing_to_do = 'X'.
ELSE.

DESCRIBE TABLE t_idoc_control_all LINES t_idoc_control_all_max.

LOOP AT t_idoc_control_all INTO s_idoc_control.

APPEND s_idoc_control TO t_idoc_control.

APPEND s_idoc_control TO t_idoc_control.

ENDIF.
ELSEIF NOT resulting_idocs[] IS INITIAL.

*** BEGIN MDG Enhancement
" Select IDOC data from database
PERFORM read_idocs

ENDIF.
ENDIF.
ENDIF.
ENDIF.
ENDIF.

IF NOT nothing_to_do IS INITIAL.
    MESSAGE i139.
ELSEIF NOT resulting_idocs[] IS INITIAL.

*** BEGIN MDG Enhancement
" Select IDOC data from database
PERFORM read_idocs

ENDIF.
ENDIF.
" Call DRF inboung for ALEAUD idocs
PERFORM drf_idoc_input_aleaud_process
   USING   gt_idoc_contrl
           gt_idoc_data
   CHANGING gt_idoc_status.

" Set IDOC status
PERFORM set_idoc_status
   USING gt_idoc_status.
*** End MDG Enhancement

CALL FUNCTION 'DB_COMMIT'.
CALL FUNCTION 'DEQUEUE_ALL'.
COMMIT WORK.
PERFORM output_list.
ELSE.
  MESSAGE i158.
*   Für die selektierten Daten gibt es keine Interessenten
ENDIF.

AT SELECTION-SCREEN ON s_upddat.
  IF sy-batch IS INITIAL AND
     ( sscrfields-ucomm = 'ONLI' OR sscrfields-ucomm = 'PRIN' ).
  IF s_upddat[] IS INITIAL.
    MESSAGE e142.
*    Es muß ein Änderungszeitraum angegeben werden.
  ENDIF.
  ENDIF.

*&---------------------------------------------------------------------*
*&      Form TIME_INTERVAL_GET
*&---------------------------------------------------------------------*
*   text
*  **********************************************************************
*       <--P_LEFT_DATE  text                                            *
* <-P_LEFT_TIME  text                      *
* <-P_RIGHT_DATE text                     *
* <-P_RIGHT_TIME text                     *
*-----------------------------------------------------------------------------------*
FORM time_interval_get
  CHANGING
    left_date LIKE edidc-upddat
    left_time LIKE edidc-updtim
    right_date LIKE edidc-upddat
    right_time LIKE edidc-updtim.

  DATA:
    job_name LIKE tbtcjob-jobname,
    job_count LIKE tbtcjob-jobcount,
    job_head LIKE tbtcjob,
    cmp_time LIKE edidc-updtim.

  CALL FUNCTION 'GET_JOB_RUNTIME_INFO'
    IMPORTING
      jobcount = job_count
      jobname = job_name.

  CALL FUNCTION 'BP_JOB_READ'
    EXPORTING
      job_read_jobcount = job_count
      job_read_jobname  = job_name
      job_read_opcode   = 19
    IMPORTING
      job_read_jobhead  = job_head.
    left_time = job_head-sdltime - 300.  " 300 seconds = 5 minutes
    cmp_time = -300.                           "23:55
    IF left_time < cmp_time.
    left_date = job_head-slddate.
    ELSE.
    *  date has to changed
    left_date = job_head-slddate - 1.
    ENDIF.
cmp_time = -240.
"23:56
IF right_time < cmp_time.
   right_date = job_head-strtdate.
ELSE.
*       date has to be changed
   right_date = job_head-strtdate - 1.
ENDIF.
ENDFORM. " TIME_INTERVAL_GET
*&---------------------------------------------------------------------*
*&       Form IDOC_CREATE
*&---------------------------------------------------------------------*
*     text
*---------------------------------------------------------------------*
* --> p1    text
* <-- p2    text
*---------------------------------------------------------------------*
FORM idocs_create
TABLES
   resulting_idocs STRUCTURE bdidocs
USING
   idoc_controls TYPE audit_idoc_control_tab.

CONSTANTS:
   c_filter_mestyp LIKE tbd10-objtype VALUE 'MESTYP'.

DATA: idoc_control TYPE audit_idoc_control_t,
       receiver_input LIKE bdi_logsys OCCURS 0 WITH HEADER LINE,
       receiver_output LIKE bdi_logsys OCCURS 0 WITH HEADER LINE,
       filter_objects LIKE bdi_fltval OCCURS 0 WITH HEADER LINE,
       to_send TYPE c,
       idoc_info TYPE audit_idoc,
       control_package TYPE audit_idoc_tab.

LOOP AT idoc_controls INTO idoc_control.
   AT NEW mestyp.
   REFRESH receiver_input.
receiver_input-logsys = idoc_control-sndprn.
APPEND receiver_input.

REFRESH filter_objects.
* append entry for filter object MESTYP
filter_objects-objtype = c_filter_mestyp.
filter_objects-objvalue = idoc_control-mestyp.
APPEND filter_objects.

CALL FUNCTION 'ALE_MESTYPE_GET_RECEIVER'
EXPORTING
message_type = c_mestyp_aleaud
TABLES
receiver_input = receiver_input
receivers = receiver_output
filterobject_values = filter_objects.

IF receiver_output[] IS INITIAL. "MDG Enhancement
APPEND LINES OF receiver_input TO receiver_output. "MDG Enhancement
ENDIF. "MDG Enhancement
IF NOT receiver_output[] IS INITIAL.
to_send = 'X'.
ELSE.
CLEAR to_send.
ENDIF.
ENDIF.
ENDAT.

IF NOT to_send IS INITIAL.
MOVE-CORRESPONDING idoc_control TO idoc_info.
APPEND idoc_info TO control_package.
ENDIF.

AT END OF sndprn.
IF NOT control_package[] IS INITIAL.
CALL FUNCTION 'AUDIT_IDOC_CREATE'
EXPORTING
rcv_system = idoc_control-sndprn
IMPORTING
  idoc_number = resulting_idocs-docnum
CHANGING
  idoc_info_records = control_package.
  exceptions
  others = 1.
REFRESH control_package.
IF NOT resulting_idocs IS INITIAL.
  APPEND resulting_idocs.
ENDIF.
ENDIF.
ENDAT.
ENDLOOP.
ENDFORM.          " IDOCS_CREATE
*&----------------------------------------------------------------------*
*&        Form output_list
*&----------------------------------------------------------------------*
*      text
*----------------------------------------------------------------------*
  -->P_RESULTING_IDOCS text
*----------------------------------------------------------------------*
FORM output_list.

header = text-001.
PERFORM fieldcat_init USING gt_fieldcat[].
PERFORM layout_init USING gs_layout.
PERFORM icon_excluding USING gs_excluding.

CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
EXPORTING
  i_callback_program = sy-repid
  i_callback_pf_status_set = g_status_set
  i_callback_user_command = g_user_command
  i_grid_title = header
  is_layout = gs_layout
  it_fieldcat = gt_fieldcat[]
it_excluding  = gs_excluding[]
TABLES
t_outtab     = resulting_idocs.

ENDFORM.          " output_list
*---------------------------------------------------------------*
*    Form  fieldcat_init                                          *
*---------------------------------------------------------------*
*   text                                                           *
*---------------------------------------------------------------*
*   -->P_GT_FIELDCAT[]    text                                    *
*---------------------------------------------------------------*

FORM fieldcat_init USING rt_fieldcat
     TYPE slis_t_fieldcat_alv.

DATA: ls_fieldcat TYPE slis_fieldcat_alv.

CLEAR ls_fieldcat.
ls_fieldcat-fieldname   = 'DOCNUM'.
ls_fieldcat-seltext_l   = text-200.
ls_fieldcat-outputlen   = '30'.
APPEND ls_fieldcat TO rt_fieldcat.

ENDFORM.           " fieldcat_init
*---------------------------------------------------------------*
*    Form  layout_init                                          *
*---------------------------------------------------------------*
*   text                                                           *
*---------------------------------------------------------------*
*   -->P_GS_LAYOUT    text                                       *
*---------------------------------------------------------------*

FORM layout_init USING rs_layout TYPE slis_layout_alv.
*doubleclick
rs_layout-f2code     = 'IDOC'.
rs_layout-colwidth_optimize  = 'X'.

SAP COMMUNITY NETWORK
© 2013-2017 SAP SE
ALE Replication using the same Client

ENDFORM. " LAYOUT_INIT
*----------------------------------------------------------------------*
* Form icon_excluding
*----------------------------------------------------------------------*
* text
*----------------------------------------------------------------------*
* -->P_GS_EXCLUDING text
*----------------------------------------------------------------------*

FORM icon_excluding USING p_gs_excluding TYPE slis_t_extab.

REFRESH p_gs_excluding[].

gs_excl_head-fcode = '&VEXCEL'. "Excel
APPEND gs_excl_head-fcode TO p_gs_excluding.

gs_excl_head-fcode = '&AQW'. "word
APPEND gs_excl_head-fcode TO p_gs_excluding.

gs_excl_head-fcode = '&GRAPH'.
APPEND gs_excl_head-fcode TO p_gs_excluding.

gs_excl_head-fcode = '&XXL'.
APPEND gs_excl_head-fcode TO p_gs_excluding.

gs_excl_head-fcode = '&CRBATCH'.
APPEND gs_excl_head-fcode TO p_gs_excluding.

gs_excl_head-fcode = '&CRTEMPL'.
APPEND gs_excl_head-fcode TO p_gs_excluding.

gs_excl_head-fcode = '&XINT'.
APPEND gs_excl_head-fcode TO p_gs_excluding.

gs_excl_head-fcode = '&URL'.
APPEND gs_excl_head-fcode TO p_gs_excluding.

gs_excl_head-fcode = '&CRDESIG'.
APPEND gs_excl_head-fcode TO p_gs_excluding.
gs_excl_head-fcode = '&VLOTUS'.
APPEND gs_excl_head-fcode TO p_gs_excluding.

gs_excl_head-fcode = '&VCRYSTAL'.
APPEND gs_excl_head-fcode TO p_gs_excluding.

gs_excl_head-fcode = '&OL0'.
APPEND gs_excl_head-fcode TO p_gs_excluding.

gs_excl_head-fcode = '&XPA'.
APPEND gs_excl_head-fcode TO p_gs_excluding.

gs_excl_head-fcode = '&OMP'.
APPEND gs_excl_head-fcode TO p_gs_excluding.

gs_excl_head-fcode = '&ILT'.
APPEND gs_excl_head-fcode TO p_gs_excluding.

ENDFORM.                    " ICON_EXCLUDING

*---------------------------------------------------------------------
* &       Form PF_STATUS_SET
*---------------------------------------------------------------------
*   text
*---------------------------------------------------------------------
*   -->> p1  text
*   <<- p2   text
*---------------------------------------------------------------------
FORM pf_status_set USING rt_extab TYPE slis_t_extab.
  DATA: l_status LIKE sy-pfkey VALUE 'STANDARD'.
  * EXCLUDING FCODES GIVEN BY ABAP LISTVIEWER *
  SET PF-STATUS l_status EXCLUDING rt_extab.
ENDFORM.                    " PF_STATUS_SET
*---------------------------------------------------------------------
FORM user_command USING rf_ucomm LIKE sy-ucomm
rs_selfield TYPE slis_selfield.

CASE rf_ucomm.
WHEN 'IDOC'.
   READ TABLE resulting_idocs INDEX rs_selfield-tabindex.
   IF sy-subrc = 0.
      SUBMIT idoc_tree_control WITH docnum = resulting_idocs-docnum
      AND RETURN.
   ELSE.
      MESSAGE s010.
      Bitte Cursor richtig positionieren
      ENDIF.
   CLEAR rf_ucomm.
WHEN OTHERS.
ENDCASE.
ENDFORM. " USER_COMMAND

FORM read_idocs
USING it_idoc_list TYPE tyt_idoc_list
CHANGING et_idoc_contrl TYPE tyt_idoc_contrl
et_idoc_data   TYPE tyt_idoc_data.

DATA:
  lv_docnum       TYPE edidc-docnum,
  ls_idoc_contrl  TYPE edidc,
  lt_idoc_data    TYPE tyt_idoc_data.

CLEAR et_idoc_contrl.
CLEAR et_idoc_data.

LOOP AT it_idoc_list INTO lv_docnum.
  CLEAR lt_idoc_data.
  CALL FUNCTION 'EDI_DOCUMENT_OPEN_FOR_READ'
    EXPORTING
      document_number = lv_docnum
    IMPORTING
      idoc_control    = ls_idoc_contrl
    EXCEPTIONS
      OTHERS          = 4.
  IF sy-subrc = 0.
    APPEND ls_idoc_contrl TO et_idoc_contrl.
  CALL FUNCTION 'EDI_SEGMENTS_GET_ALL'
    EXPORTING
      document_number         = lv_docnum
      idoc_containers         = lt_idoc_data
    EXCEPTIONS
      document_number_invalid = 1
      end_of_document         = 2
      OTHERS                  = 3.
  IF sy-subrc <> 0.
    MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno
      WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
  ELSE.
APPEND LINES OF lt_idoc_data TO et_idoc_data.
ENDIF.
ENDIF.
CALL FUNCTION 'EDI_DOCUMENT_CLOSE_READ'
EXPORTING
document_number = lv_docnum.
ENDLOOP.
ENDFORM.                       "read_idocs

*---------------------------------------------------------------------*
* Form set_idoc_status
*---------------------------------------------------------------------*
* text
*---------------------------------------------------------------------*
* -->IT_IDOC_STATUS  text
*---------------------------------------------------------------------*
FORM set_idoc_status
       USING it_idoc_status TYPE tyt_idoc_status.

DATA:
  ls_idoc_status      TYPE bdidocstat,
  lt_idoc_data        TYPE tyt_idoc_data,
  ls_edi_status_record TYPE LINE OF tyt_edi_status_records,
  lt_edi_status_records TYPE tyt_edi_status_records.

LOOP AT it_idoc_status INTO ls_idoc_status.
CALL FUNCTION 'EDI_DOCUMENT_OPEN_FOR_EDIT'
EXPORTING
document_number = ls_idoc_status-docnum
* ALREADY_OPEN = 'N'
* IMPORTING
* IDOC_CONTROL =
TABLES
  idoc_data = lt_idoc_data
EXCEPTIONS
document_foreign_lock = 1
document_not_exist = 2
document_not_open = 3
status_is_unable_for_changing = 4
OTHERS = 5

IF sy-subrc <> 0.
* Implement suitable error handling here
ENDIF.

CLEAR ls_edi_status_record.
CLEAR lt_edi_status_records.
MOVE-CORRESPONDING ls_idoc_status TO ls_edi_status_record.
ls_edi_status_record-mandt = sy-mandt.
ls_edi_status_record-logdat = sy-datum.
ls_edi_status_record-logtim = sy-zeit.
ls_edi_status_record-uname = sy-uname.
ls_edi_status_record-status = ls_idoc_status-status.
ls_edi_status_record-repid = sy-repid.
APPEND ls_edi_status_record TO lt_edi_status_records.
CALL FUNCTION 'EDI_DOCUMENT_CLOSE_EDIT'
EXPORTING
  document_number = ls_idoc_status-docnum
*  DO_COMMIT = 'X'
*  DO_UPDATE = 'X'
*  WRITE_ALL_STATUS = 'X'
*  STATUS_75 = ''
TABLES
  status_records = lt_edi_status_records
EXCEPTIONS
  idoc_not_open = 1
  db_error = 2
  OTHERS = 3.
IF sy-subrc <> 0.
* Implement suitable error handling here
ENDIF.

ENDLOOP.
ENDFORM. "set_idoc_status

*&----------------------------------------------------------------------*
*&     Form drf_idoc_input_aleaud_process
*&----------------------------------------------------------------------*
*   text
*&----------------------------------------------------------------------*
*   -->>IT_IDOC_CONTRL text
*   -->>IT_IDOC_DATA  text
*   -->>ET_IDOC_STATUS text
*&----------------------------------------------------------------------*
FORM drf_idoc_input_aleaud_process
  USING  it_idoc_contrl  TYPE tyt_idoc_contrl
      it_idoc_data   TYPE tyt_idoc_data
  CHANGING et_idoc_status  TYPE tyt_idoc_status.

DATA:
lt_idoc_status          TYPE TABLE OF bdidocstat,
lt_return_variables     TYPE TABLE OF bdwfretvar,
lt_serialization_info   TYPE TABLE OF bdi_ser.

CLEAR et_idoc_status.

" Call DRF inbound for ALEAUD idocs
CALL FUNCTION 'DRF_IDOC_INPUT_ALEAUD'
  EXPORTING
      input_method         = space
      mass_processing      = space
  TABLES
      idoc_contrl          = it_idoc_contrl
      idoc_data            = it_idoc_data
      idoc_status          = et_idoc_status
      return_variables     = lt_return_variables
      serialization_info   = lt_serialization_info
  EXCEPTIONS
wrong_function_called = 1
OTHERS = 2.

ASSERT sy-subrc = 0.

ENDIF. "drf_idoc_input_aleaud_process
**Copyright**

© Copyright 2013-2017 SAP SE. 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, Excel, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, System i, System i5, System p, System p5, System x, System z, System z9, z10, z9, iSeries, pSeries, xSeries, zSeries, eServer, z/VM, z/OS, i5/OS, S/390, OS/390, OS/400, AS/400, S/390 Parallel Enterprise Server, PowerVM, Power Architecture, POWER6+, POWER6, POWER5+, POWER5, POWER, OpenPower, PowerPC, BatchPipes, BladeCenter, System Storage, GPFS, HACMP, RETAIN, DB2 Connect, RACF, Redbooks, OS/2, Parallel Sysplex, MVS/ESA, AIX, Intelligent Miner, WebSphere, Netfinity, Tivoli and Informix are trademarks or registered trademarks of IBM Corporation.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

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 Oracle Corporation.

JavaScript is a registered trademark of Oracle Corporation, used under license for technology invented and implemented by Netscape.

SAP, R/3, SAP NetWeaver, Duet, PartnerEdge, ByDesign, SAP Business ByDesign, 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 other countries.

Business Objects and the Business Objects logo, BusinessObjects, Crystal Reports, Crystal Decisions, Web Intelligence, Xcelsius, and other Business Objects products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Business Objects S.A. in the United States and in other countries. Business Objects is an SAP company.

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.