SAP How-To Guide: Extend the MDG BP – Ensure Auto-creation of ERP Vendors with MDG-S

Applies to:

Master Data Governance for Business Partners (MDG-BP), Customers (MDG-C) and Suppliers (MDG-S) with release versions SAP ERP 6 Enhancement Package 6 and newer. For more information, visit the Master Data Management homepage (https://www.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 business partner, customer and supplier data and their related governance solutions.

Author(s): SAP SE

Company: SAP SE

Created on: September 2018

Version: 1.2

Table of Contents

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.

Domain-specific content (data models, user interfaces, workflows) is provided as part of the standard for several application areas. It is a common requirement from customers to adapt the MDG data models to their specific needs.

This document explains the extension of the MDG Business Partner / Customer / Supplier solution to ensure that ERP Vendor records are always created when users create suppliers in the main user interface for Supplier Governance.

The given scenario is rather complex. It requires a custom handler classes and a specific implementation of the data derivation. Before following the instructions in this guide, we recommend you study the following guides:

- Extensibility Options for SAP Master Data Governance
    - How-to Guide: Extending the MDG Business Partner – Overview
    - How-to Guide: Create and Register a Custom Handler Class

Scenario

The SAP delivered MDG-S UI configuration is designed in a way that allows the creation of a supplier with or without an ERP vendor record. This supports the different use cases of the supplier. An SRM supplier does not need the ERP vendor data.

The requirement is to change the behavior of the existing MDG-S UI so that it always creates the ERP vendor record, even if the user does not maintain any ERP vendor-specific data (such as general data, company codes or purchasing organizations).

Technical Background

In the SAP delivered MDG-S UI configuration, the creation of the ERP vendor is triggered by pressing the ***New*** button on the ***ERP Vendors*** list. The first vendor created is the so-called **standard assignment**. The standard vendor assignment has a fixed assignment category **SUPPL** (this category is fixed for all vendor assignments) and a fixed assignment ID **000000000001**.

Since the task is the automated creation of the standard vendor assignment, a suitable place to implement a solution is the data derivation functionality of a handler class. In that case, it makes sense to create a customer handler class that inherits from the abstract MDG application handler since the objects to be created belong to this software layer. The custom derive functionality needs to create both a multiple assignment entity and an ERP-specific vendor general data entity. A suitable trigger could be the creation of a business partner entity since this happens only once right during the start of the UI application.

Step by Step Implementation Guide

The following steps provide details how-to ensure the creation of an ERP vendor record.

1. Follow the steps of how-to guide Create and Register a Custom Handler Class to create your own handler class by inheriting from class **CL_MDG_BS_ECC_HANDLER**. An existing custom handler can be reused, of course.
2. Get familiar with the data derivation implementations of the multiple assignments' handler class **CL_MDG_BS_MLT_ASSGNMNT_HANDLER** and the supplier's handler class **CL_MDG_BS_SUPPL_HANDLER**. Do you notice that both classes basically already provide the application logic that you need for the scenario and you only need to combine the classes in a different way?
3. As mentioned before, the trigger for creating the ERP vendor is the creation of the business partner during the start of the application. Since the implementation of the multiple assignments' handler method **BUFFER_DERIVED_DATA** already investigates creations of the business partner entity and stores this trigger in the attribute **GT_BPROOT_TASK**, there is no need to do this again in the custom handler.
4. Go to the custom handler class and select the redefinition of method **DERIVE_DATA**. This is the correct place for the creation of the ERP vendor. Your coding needs to:
   a. Ensure that it is only executed for the creation of a single, new business partner of type supplier.
   b. Double-check the derive buffer to ensure that the derivation has not happened before by any other handler class or rule service derivation.
   c. Prepare the multiple assignment entity type.
   d. Prepare the ERP Vendor’s general data entity type.
   e. Derive both including the update of the local derive buffer.
5. (See the appendix for the complete sample coding.)
6. **Save** and **Activate** the method.

Result

The derivation functionality of the custom handler class now creates the ERP vendor records automatically as soon as the UI is started. Run the user interface application for Supplier Governance to prove this. As soon as the UI loaded, an entry in the list **ERP Vendors** is displayed. This record is the derived standard supplier assignment. Navigate into the edit page to check the derived ERP vendor general data values such as the **Account Group**.

Additional Information

Links

Customer Vendor Integration on SAP Help

FPM on SCN

- FPM Cookbook

MDG Guides on Service Market Place

How-to Guides

Extensibility Options for SAP Master Data Governance

- How-to Guide: Extending the MDG Business Partner – Overview
- How-to Guide: Create and Register a Custom Handler Class

SAP Notes

- 1637249 specifying required information for OSS support
- 2105467 specifying required information for Performance Issues
- 2221398 providing a detailed overview of all fields in MDG-BP.

Version History

- 1.1 – Updated Formatting & Links
- 1.2 – Refactoring according to the latest SAP MDG, Central Governance release

Appendix

```
  METHOD if_mdg_bs_bp_access_handler~derive_data.
*! The example shows how to automatically create an ERP Vendor entity.
*
*  It respects the actual hierarchy of entity types by also creating
*  required Multiple Assignment entity type which is the leading parent
*  of an ERP Vendor.
*
*  IMPORTANT NOTE
*  Using the derivation for CREATING entities can cause issues if you
*  also use the BOL Entity Cloner (Copy) functionality. You have to be
*  aware that derivation can create objects that would also be copied.
*  This may lead to error messages, such as "There is already an object
*  with the same key information". If so, you need to decide whether you
*  want to derive or to copy. For more information about how-to control
*  the BOL copy, see the appendix of SAP note 2020896.

    "Since the custom handler is called for all UI configurations and
    "various business activities (like create, change, initial load),
    "it is recommended to limit the automated creation to
    "a) the MDG-S related business activity for creating a supplier
    "b) the exact amount of one single record
    IF mv_process <> 'SUP1'
      OR lines( gt_bproot_task ) <> 1.
      RETURN.
    ENDIF.

    "Furthermore we want to create the ERP Vendor only once during the
    "start-up of the UI
    FIELD-SYMBOLS <ls_bproot_task> LIKE LINE OF gt_bproot_task.
    READ TABLE gt_bproot_task ASSIGNING <ls_bproot_task> INDEX 1.
    IF sy-subrc <> 0
      OR <ls_bproot_task>-task <> gc_ins.
      RETURN.
    ENDIF.

    "Add the creation of the multiple assignment entity for a standard
    "supplier. The coding is similar to the one in the multiple
    "assignments' handler method DERIVE_STANDARD_ASSIGNMENT.
    DATA:
      ls_bp_mlt_as TYPE /mdgbpx/_s_bp_pp_bp_mlt_as,
      lt_bp_mlt_as TYPE tty_bp_mlt_as_sorted.
    FIELD-SYMBOLS <ls_mlt_as_derived_data> LIKE LINE OF gt_mlt_as_derived_data.

    "Check: assignment is not in buffer
    READ TABLE gt_mlt_as_derived_data ASSIGNING <ls_mlt_as_derived_data>
      WITH KEY partner-header-object_instance-bpartner = <ls_bproot_task>-bp_id.
    IF sy-subrc EQ 0.
      READ TABLE <ls_mlt_as_derived_data>-mlt_assignments TRANSPORTING NO FIELDS
        WITH KEY assignment_id = if_mdg_bs_ecc_bp_constants=>gc_sp_std_mlt_assig
nment_id
                 partner       = <ls_bproot_task>-bp_id.
      IF sy-subrc EQ 0.
        RETURN.
      ENDIF.
    ENDIF.

    "Prepare assignment data
    ls_bp_mlt_as-
assgnm_id = if_mdg_bs_ecc_bp_constants=>gc_sp_std_mlt_assignment_id.
    ls_bp_mlt_as-bp_header = <ls_bproot_task>-bp_id.
    ls_bp_mlt_as-
as_type   = if_mdg_bs_ecc_bp_constants=>gc_sp_mlt_assignment_cat.
    ls_bp_mlt_as-standard  = abap_true.

    "Add the creation of the ERP vendor general data entity for a
    "standard supplier next.  The coding is similar to the one in the
    "supplier's handler method DERIVE_SUPPLIER.
    DATA:
      lo_exception TYPE REF TO cx_mdg_bs_ecc_bp_obj_integr,
      ls_bp_vengen TYPE /mdgbpx/_s_bp_pp_bp_vengen,
      ls_message   TYPE usmd_s_message,
      lt_bp_vengen TYPE tty_bp_vengen_sorted,
      lv_bp_group  TYPE bu_group.

    "Check: general data is not in buffer (inbound!)
    IF <ls_mlt_as_derived_data> IS ASSIGNED.
      READ TABLE <ls_mlt_as_derived_data>-vendors TRANSPORTING NO FIELDS
        WITH KEY assignment_id = if_mdg_bs_ecc_bp_constants=>gc_sp_std_mlt_assig
nment_id.
      IF sy-subrc EQ 0.
        RETURN.
      ENDIF.
    ENDIF.

    "Prepare supplier entity keys
    ls_bp_vengen-bp_header = <ls_bproot_task>-bp_id.
    ls_bp_vengen-
assgnm_id = if_mdg_bs_ecc_bp_constants=>gc_sp_std_mlt_assignment_id.

    "Derive account group and ID.
    me->derive_bp_group(
      EXPORTING
        io_model        = io_model
        iv_bp_id        = <ls_bproot_task>-bp_id
      IMPORTING
        ev_bp_group     = lv_bp_group
      CHANGING
        ct_message_info = ct_message_info ).
    IF lv_bp_group IS INITIAL.
      "There's no (default) grouping for the BP. Without the grouping,
      "the derivation cannot proceed!
      RETURN.
    ENDIF.
    TRY.
        ls_bp_vengen-
ktokk = cl_mdg_bs_ecc_bp_services=>get_supplier_account_for_group( lv_bp_group )
.
        ls_bp_vengen-lifnr = cl_mdg_bs_ecc_bp_services=>get_supplier_number(
                               iv_account_group = ls_bp_vengen-ktokk
                               iv_bp_group      = lv_bp_group
                               iv_partner       = <ls_bproot_task>-bp_id ).
      CATCH cx_mdg_bs_ecc_bp_obj_integr INTO lo_exception.
        CLEAR ls_message.
        ls_message-msgty = 'E'.
        ls_message-msgid = lo_exception->if_t100_message~t100key-msgid.
        ls_message-msgno = lo_exception->if_t100_message~t100key-msgno.
        ls_message-msgv1 = lo_exception->msgv1.
        APPEND ls_message TO ct_message_info.
        RETURN.
    ENDTRY.

    "Derive sub-range and plant relevance.
    ls_bp_vengen-
ltsna = cl_mdg_bs_ecc_bp_services=>is_sub_range_relevant( ls_bp_vengen-ktokk ).
    ls_bp_vengen-
werkr = cl_mdg_bs_ecc_bp_services=>is_plant_level_relevant( ls_bp_vengen-
ktokk ).

    "Derive trading partner
    ls_bp_vengen-vbund = me->derive_trading_partner(
                           io_model = io_model
                           iv_bp_id = <ls_bproot_task>-bp_id ).

    "WRITE the derived data
    INSERT ls_bp_mlt_as INTO TABLE lt_bp_mlt_as.
    io_write_data-
>write_data( i_entity = if_mdg_bs_ecc_bp_constants=>gc_ma_model_entity
                               it_data  = lt_bp_mlt_as ).
    me->if_mdg_bs_bp_access_handler~buffer_derived_data(
      EXPORTING
        io_model    = io_model
        iv_entity   = if_mdg_bs_ecc_bp_constants=>gc_ma_model_entity
        it_data_ins = lt_bp_mlt_as ).
    INSERT ls_bp_vengen INTO TABLE lt_bp_vengen.
    io_write_data-
>write_data( i_entity = if_mdg_bs_ecc_bp_constants=>gc_sp_model_entity-
general_data
                               it_data  = lt_bp_vengen ).
    me->if_mdg_bs_bp_access_handler~buffer_derived_data(
      EXPORTING
        io_model    = io_model
        iv_entity   = if_mdg_bs_ecc_bp_constants=>gc_sp_model_entity-
general_data
        it_data_ins = lt_bp_vengen ).
  ENDMETHOD.
```

Copyright

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.