How to Configure Data Replication for MDG Custom Objects (Flex Option)

Applicable Releases:
EhP5

Version 1.0
September 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>Example Text</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>Example text</td>
<td>Emphasized words or phrases in body text, graphic titles, and table titles</td>
</tr>
<tr>
<td>Example text</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>Example text</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>&lt;Example text&gt;</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>EXAMPLE TEXT</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........................................................................................................... 2

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

3. Prerequisites .................................................................................................................... 2

4. Step-by-Step Procedure: ............................................................................................... 3
   4.1 Create the ZZ Data Model Using the Flex Option .................................................. 3
   4.2 Enable Key Mapping (Optional Step) ...................................................................... 6
   4.3 Create a Business Object Type ............................................................................... 7
   4.4 Define Object Nodes .............................................................................................. 8
   4.5 Create and Assign Object Identifier Types ............................................................. 9
   4.6 Generate Structures ............................................................................................... 10
   4.7 Assign a Key Structure to Object Identifier Types ............................................... 12
   4.8 Assign a Data Model to a Business Object Type .................................................... 12
   4.9 Create an Outbound Interface ............................................................................... 13
   4.10 Create an Outbound Implementation ................................................................... 17
   4.11 Create a Filter Object .......................................................................................... 18
   4.12 Assign a Filter to a Business Object Type or an Outbound Implementation .......... 19
   4.13 Create a Replication Model .................................................................................. 20
   4.14 Create Filter ......................................................................................................... 21
   4.15 Replicate Data ....................................................................................................... 22
1. Business Scenario

Data Replication from your customer specific Data Model (Flex Option).

2. Background Information

MDG offers the following options to store active master data (data that has been approved):

- The **reuse option** used by MDG-M and MDG-S stores data in the SAP ERP tables such as MARA or LFA1.
- The **flex option** used by MDG-F and MDG for Custom Objects stores data in generated tables.

In both options, inactive master data (data that has not yet been approved) is stored in the generated tables.

Data that the MDG system replicates to target systems is always active data. The MDG system takes the active data from the SAP ERP tables or from the generated tables depending on the option in use (reuse option or flex option).

MDG applications such as MDG-M, MDG-S, and MDG-F include standard implementations of the Data Replication Framework (DRF) that read the data and send the messages to the target system. The standard implementations support key mapping and value mapping.

SAP also delivers generic implementations that you can configure to replicate data from customer-specific applications (MDG for Custom Objects). This guide describes the necessary configuration steps.

You can perform most configuration tasks in Customizing for Master Data Governance under SAP Reference IMG -> Cross Application Components -> Processes and Tools for Enterprise Applications - > Master Data Governance.

Additionally, you can use the following transactions:

- **MDGIMG** – IMG Master Data Governance
- **DRFIMG** – IMG Data Replication Framework
- **IDMIMG** – IMG Key Mapping
- **VMIMG** – IMG Value Mapping

3. Prerequisites

This guide shows the configuration steps using a small example. The first section briefly explains the configuration of the Data Model.
4. Step-by-Step Procedure:

4.1 Create the ZZ Data Model Using the Flex Option

This example involves a simple definition of the Data Model that includes two Entity Types and a relationship. The example covers a small part of the SFLIGHT scenario, a scenario that is often used in SAP training materials.

The assumption behind the example is that you are building a custom MDG application for creating and governing data about airlines.

In Customizing for Master Data Governance (transaction MDGIMG), choose General Settings -> Data Modeling -> Edit Data Model. In the Entity Types view, edit the CARR Entity Type for Carrier (Airline) as shown below.

The Entity Types view of the General Settings -> Data Modeling -> Edit Data Model Customizing activity.

The CARR Entity Type.

Specify an attribute URL for the CARR Entity Type.
How to Configure Data Replication for MDG Custom Objects (Flex Option)

The Attributes view

In the Entity Types view, edit the CURR Entity Type for Currency as shown below.

The CURR Entity Type.

In the Relationships view, specify the relationship between CARR and CURR.

After you generate the Data Model, you can use the USMD_DATA_MODEL report to identify the tables generated for your Data Model. You can access this report from transaction SE38.
Generated tables shown after running the **USMD_DATA_MODEL** report from transaction **SE38**.
4.2 Enable Key Mapping (Optional Step)

This step is required if you want to implement key mapping.

<table>
<thead>
<tr>
<th>Key Mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Overall Information</td>
</tr>
<tr>
<td>• Customize Business Objects for Key Mapping</td>
</tr>
<tr>
<td>• Enhance Key Mapping Content</td>
</tr>
</tbody>
</table>

- Define Business Objects
- Define Object Identifiers
- Assign Key Structures to Object Identifiers
- Define Object Nodes
- Assign Business Objects to Main Contexts

- BADI: Determination of Local System Name
  Define a Mapping Context for UKMS

**Transaction** IDMIMG: Define a Mapping Context for UKMS -> Define Mapping Contexts.

**New Entries: Overview of Added Entries**

<table>
<thead>
<tr>
<th>UKM: Mapping context</th>
<th>Subcontext</th>
<th>Main Cnxt</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZZSFLIGHT</td>
<td>SAPdefault:Mapping</td>
<td>ZZSF</td>
</tr>
</tbody>
</table>

The system generates a set of tables based on standard tables. These tables should have a prefix of Z to identify them as customer specific objects.

**Copy Table**

From
Table: `UKMDB_AGC00000`
to
Table: `ZUKMDB_AGCZZSF0`

The system requires confirmation of tables to be copied.
4.3 Create a Business Object Type

Data replication always refers to business object types, which are based on data models. You can define business object types in the Define Business Objects customizing activity or in the Define Business Objects Available for Replication customizing activity.

Transaction IDMIMG

For key mapping, you must assign each Business Object Type to a Main Context. In this example the ZZSF Business Object Type is assigned to the ZZSFLIGHT Main Context.

Transaction IDMIMG
4.4 Define Object Nodes

Define the **ZZSF Object Node Type** for the **ZZSF** airline code.

- **Key Mapping**
  - Define Business Objects
  - Define Object Identifiers
  - Assign Key Structures to Object Identifiers

Transaction **IDMIMG**

<table>
<thead>
<tr>
<th>Ob Node Ty</th>
<th>Obi. Node Type Desc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZZSF</td>
<td>ZZSF Airline Code</td>
</tr>
</tbody>
</table>

You can use the **Define Object Identifiers** customizing activity if there are different **Object Identifier Types** that must map to each other (for example, if a GUID must map to a number.)

An example used for the business partner is shown below.

Transaction **IDMIMG**: **Enhance Key Mapping Content -> Define Object Identifiers.**
4.5 Create and Assign Object Identifier Types

Create the **ZZSF Object Identifier Type**. When doing this, specify the **ZZSF Object Node Type** defined earlier.

Transaction **IDMIMG**

Next, assign the **ZZSF Object Identifier Type** to the **ZZSF Business Object Type**.

Transaction **IDMIMG**

Fields in the **Define Business Objects** Customizing activity.
4.6 Generate Structures

Create a package in SE80.

Create a package in SE80.

[Image]

Object Name | Description
---|---
ZZ_DRF | Custom Object SFLIGHT Data Replication

Generate data model-specific structures for the **CARR Entity Type**.

Transaction **MDGIMG: General Settings -> Data Modeling**

<table>
<thead>
<tr>
<th>Entity Type</th>
<th>Where Used</th>
<th>Prefix/Namespace</th>
<th>Name of Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>CARR</td>
<td>DRF Structures</td>
<td>ZXX</td>
<td>ZDRF_CARR</td>
</tr>
</tbody>
</table>

Elements of the **Generate Data Model-specific Structures** Customizing activity.

Confirmation message after the activation of data model-specific structure.

Do the same for the **CURR Entity Type**.
Use transaction SE11 to check the structures for the **ZXX_SZZ_ZDRF_CARR** Data type.

Searching for data model-specific structure using transaction **SE11**.

Whenever MDG generates these structures, it activates them.
4.7 Assign a Key Structure to Object Identifier Types

Assign a Key Structure to Object Identifier Types

Transaction DRFIMG

Elements of the Assign Key Structures to Object Identifier Customizing activity.

4.8 Assign a Data Model to a Business Object Type

Assign a Data Model to a Business Object Type

Transaction MDGIMG: General Settings -> Data Modeling -> Edit Data Model.

Elements of the Edit Data Model Customizing activity.

Generate the Data Model.
4.9 Create an Outbound Interface

To prepare for the creation of an outbound interface, run transaction SE80 (Create Package / Function Group).

Create the `Z_ZZ_PACKAGE` package.

Create the `Z_ZZ_FUNC_GROUP` function group in the `Z_ZZ_PACKAGE` package.

Generate an Outbound Interface Model either in Customizing or using transaction OIF_MAINTAIN.

Transaction DRFIMG (Also available using transaction OIF_MAINTAIN.)
How to Configure Data Replication for MDG Custom Objects (Flex Option)

Create and define an Interface Model ID.

**Create Outbound Interface Model**

Create and define an Interface Model ID.

**Create Outbound Interface Model**

Enter header data for the interface model and resulting function module

- **Interface Model ID:** ZZ_SFLIGHT
- **Interface Model Description:** SFlight Outbound Model (ZZ)
- **Object Type Code:** ZZSF
- **Package Name:** 2_ZZ_PACKAGE
- **Function Group Name:** Z_ZZ_FUN_GROUP
- **Name:** ZZ_SFLIGHT
- **Description:** Generated RFC for SFlight Outbound Model (ZZ)

Choose the Name ABAP Dictionary Objects pushbutton to define and name structures.

**Create Outbound Interface Model : ZZ_SFLIGHT**

Select the entity types and attributes you want to include in an interface model. Then enter names for resulting dictionary objects. Use the naming function provided to ensure a consistent naming scheme.

**Select Entities and Attributes**

- **Name:** Z
  - **Include in Interface Model:** [✓]
  - **Structure Name:** ZZSF_Z_CARR
  - **Structure Description:** Structure for Z_CARR
  - **Table Type Name:** ZZSF_T_CARR
  - **Table Type Description:** Table Type for Z_CARR

- **Name:** CARR - Airline
  - **View Attributes:** [✓]
  - **CARR_CURREN - Airline currency:** [✓]
  - **CARR_CURREN - Description (medium text):** [✓]
  - **CARR_T - Travel agency URL:** [✓]
How to Configure Data Replication for MDG Custom Objects (Flex Option)

The Transport Request Manager that opens when you complete Step 3 in creating an Outbound Interface Model – Review and Submit. You can use the same transport to transfer the function module in the target system later on.

The Application Log that opens in Step 4 of creating an Outbound Interface Model – Create Application Log.
After you define the **ZZ_SFLIGHT Outbound Interface Model**, the system generates the **ZZ_SFLIGHT** function module (see below). The outbound implementation defined in the DRF calls this function model.

### The ZZ_FLIGHT function module

```plaintext
FUNCTION ZZ_SFLIGHT.

**Local Interface:**

**IMPORTING**

**VALUE(IV_CARR) TYPE I3SF_T_CARR**

**VALUE(IV_RFC_DEST) TYPE RFC_DEST**

**EXPORTING**

**VALUE(ET_MESSAGE) TYPE BAFINET2_T**

**VALUE(ER_CARR) TYPE SY_SUBRC**

**EXCEPTIONS**

**SYSTEM_FAILURE**

**COMMUNICATION_FAILURES**

**This is a generated FM by DDF.**

**Please do not change the signature of the FM.**

**By default it calls the same FM with an RFC destination assuming**

**that the FM with the same name would exist in the receiver system.**

**You can reimplement the FM by commenting out the call to the FM**

**and then implement the function in your respective inbound FM based on the destination.**

**CALL FUNCTION 'ZZ_SFLIGHT'**

**DESTINATION IV_RFC_DEST**

**IMPORTING**

**IT_CARR**

**IV_RFC_DEST** = IV_RFC_DEST

**ET_MESSAGE** = ET_MESSAGE

**ER_CARR**

**EXCEPTIONS**

**SYSTEM_FAILURE** = *

**COMMUNICATION_FAILURE** = *

**IF SY_SUBRC <> 0. ER_CARR = 0. ENDIF.**

ENDFUNCTION.
```
4.10 Create an Outbound Implementation

Transaction DRFIMG

When defining an outbound implementation, use the generic outbound implementation class (CL_MDG_OIF_DRF_OUTBOUND_IMPL). You can copy this class to allow additional capabilities that are not supported by default such as key mapping and value mapping. For more information, you can refer to the standard outbound implementations that SAP delivers for other objects.

For your outbound implementation, you must specify a Business Object Type (in this case ZZSF_01), an Outbound Implementation Class (in this case CL_MDG_OIF_DRF_OUTBOUND_IMPL), and an Outbound Interface Model ID (in this case ZZ_SFLIGHT). The Outbound Interface Model ID is the last column in a table that requires scrolling. The two screenshots below show the relevant fields.
4.11 Create a Filter Object

Create a Filter Object.

Transaction DRFIMG

**Change View "Define Filter Objects": Overview**

Fields in the Define Filter Objects Customizing activity.

Leave the Table Name field blank.

- **Note**

  A complex filter such as the one in the example does not require a table name. The system only requires table names for simple filters. Such filters are only available for standard applications that are built using the reuse option.

If required, you can define your own structure to include all relevant fields from the generated table. In the Assign Filters view, apply the following settings.

  - For the Filter field, use codes between 80 and 99. This range is assigned to the customer namespace.
  - Use the generic Filter Class CL_MDG_OIF_DRF_FILTER.

You do not have to assign an Entity Type.
### 4.12 Assign a Filter to a Business Object Type or an Outbound Implementation

You can assign a *Filter Object* to a *Business Object Type*.

Transaction **DRFIMG**

Alternatively, you can assign a *Filter Object* to an outbound implementation, which is more specific than a *Business Object Type*.

Transaction **DRFIMG**

Fields in the *Define Outbound Implementations* Customizing activity.
4.13 Create a Replication Model

Define the ZZSF Replication Model. This customizing activity is client-specific.
Run transaction DRFIMG and choose Define Custom Settings for Outbound Implementations -> Define Replication Models.

The Data Model field is specific to MDGF. You do not need to specify a Data Model for custom objects.

Assign the ZZSF_01 outbound implementation to the ZZSF Replication Model.

Assign the Business System or systems that act as receiver systems for the combination of the Replication Model and the Outbound Implementation.

Make sure you activate the Replication Model.

The Activate Replication Model button in the Define Replication Models Customizing activity.
4.14 Create a Filter

Run transaction DRFF.

The Filter Criteria screen.

Define a filter specifying that *Airline local currency* is EUR.

The Change Filter Criteria screen.
4.15 Replicate Data

Execute data replication in transaction **DRFOUT**.

**Execute Data Replication**

Replication Model

<table>
<thead>
<tr>
<th>Replication Model</th>
<th>ZZSF</th>
<th>Replication Model for SFLIGHT</th>
</tr>
</thead>
</table>

Outbound Implementation

<table>
<thead>
<tr>
<th>Outbound Implementation</th>
<th></th>
</tr>
</thead>
</table>

Replication Mode

- **Initialization**
- **Changes**

- **Limit Changes Using Time Interval**
  - Creation Date Lower Limit
  - Creation Time Lower Limit: 00:00:00
  - Creation Date Upper Limit
  - Creation Time Upper Limit: 00:00:00

- **Manual**
- **Manual Replication Filter Criteria Available**

Replication of the ZZSF *Replication Model* using the *Execute Data Replication* (**DRFOUT**) transaction.
The system calls the ZZ_SFLIGHT outbound interface.

ABAP debugger shows the system calling the ZZ_SFLIGHT outbound interface.

Objects specified as filter criteria are passed to the ZZ_SFLIGHT function module.

ABAP debugger shows the passing of data to the function module.
Note

This example ends with the data passed to the function module in the sender system. The next step is to create a function module with the same name (ZZ_SFLIGHT) in the receiver system. The function module in the sender system (shown above) calls the function module in the receiver system. Additionally, you need to implement settings that allow the posting of the data.

You can consider more sophisticated implementations like using ALE / IDocs or WebServices for the data replication. To do this, start by implementing an RFC-enabled function module in the receiver system. Based on the implementation of the function module, you can generate a BAPI and an IDoc (including functions modules for sending and receiving). The function module of the sender system (shown above) calls the function module you implement in the receiver system. Likewise, you can generate and consume a WebService.

www.sdn.sap.com/irj/sdn/howtoguides