

SAP How-To Guide: Extend the MDG Business Partner - Overview



Applies to:

Supplier Governance (MDG-S) and Customer Governance (MDG-C) running on SAP ECC 6 EhP 6 Master Data Governance and newer. For more information, visit the Master Data Management homepage.

<http://scn.sap.com/community/mdm/master-data-governance>

Summary

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

If your domain-specific solution does not fully meet requirements, you can customize and extend it. This guide provides you with the foundation knowledge you need to extend business partner data and its related governance solutions: customer governance (MDG-C) and supplier governance (MDG-S).

Authors: Michael Theis, Lars Rueter

Company: SAP AG

Created on: September 2013

Version: 1.2

Table of Contents

Introduction	4
Business Partners, Customers and Suppliers	4
Business Partner Governance (MDG-BP)	4
Customer Governance (MDG-C)	5
Supplier Governance (MDG-S)	5
Customer Vendor Integration (CVI)	6
Multiple Assignments	8
MDG Data Model and Access/Handler class concept	9
Software Layers	9
Data Derivation	9
Access Classes	10
Generic Handler Classes	10
Handler Class for Business Partner Governance (MDG-BP)	11
Handler Class for Multiple Assignments	11
Handler Class for Customer Governance (MDG-C)	11
Handler Class for Supplier Governance (MDG-S)	12
Modeling and Implementation for the Generic Interaction Layer (genIL)	12
MDG-BP	13
Multiple Assignments	14
MDG-C	15
Supplier Governance (MDG-S)	17
User Interfaces	20
Context Based Adaptations	20
MDG-BP	22
Overview	22
ABAP Object List	22
Multiple Assignments	22
Overview	22
ABAP Object List	22
Customer Governance (MDG-C)	23
Overview	23

ABAP Object List: Generic Components.....	23
ABAP Object List: Enterprise Search.....	23
ABAP Object List: General Data	23
ABAP Object List: Company Codes.....	24
ABAP Object List: Dunning Areas.....	25
ABAP Object List: Extended Withholding Tax Types	26
ABAP Object List: Sales Areas	26
ABAP Object List: Partner Functions	27
ABAP Object List: Tax Indicators.....	27
ABAP Object List: Customer-like UI.....	27
ABAP Object List: Lean Vendor Creation UI.....	28
Supplier Governance (MDG-S).....	28
Overview	28
ABAP Object List: Generic Components.....	30
ABAP Object List: Enterprise Search.....	30
ABAP Object List: General Data	30
ABAP Object List: Company Codes.....	31
ABAP Object List: Dunning Areas.....	32
ABAP Object List: Extended Withholding Tax Types	32
ABAP Object List: Purchasing Organization	32
ABAP Object List: Different Purchasing Data.....	33
ABAP Object List: Partner Functions	34
ABAP Object List: Sub-Ranges.....	34
ABAP Object List: Vendor-like UI.....	34
ABAP Object List: Lean Vendor Creation UI.....	35
Additional Information.....	36
Links	36
How-to Guides	36
SAP Notes.....	36
Copyright.....	37

Introduction

This document provides the basic information you need to extend the SAP-delivered solution for business partner within MDG. The solution for business partner incorporates customer governance (MDG-C) and supplier governance (MDG-S). In addition to explaining the key concepts and implementation details, this document links to guides that provide real-life examples.

The document is valid for MDG in the area of Business Partner starting with enhancement package 6. Some chapters contain specific information being available only in the MDG 6.1 release.

Business Partners, Customers and Suppliers

MDG offers different solutions in the business partner area.

Business Partner Governance (MDG-BP)

MDG-BP is the foundation for MDG-C and MDG-S. Each customer and supplier is based upon a business partner. It reflects the common SAP Business Partner (the one you can maintain in SAPGUI with transaction BP).

The scope of developments for business partner governance in EHP6 is as follows:

- General Data (table BUT000)
 - Only Organization Data
- Addresses
 - International Address Versions
- Address Usages
- Communication Data
 - Only address dependent communication data
 - Telephone Numbers
 - Mobile Numbers
 - Fax Numbers
 - E-Mail Addresses
 - Web Sites
- Roles
- Bank Accounts
- Identification Numbers
- Tax Numbers
- Industries

The scope of developments for business partner governance in MDG 6.1 is as follows:

- Extended General Data (table BUT000)
 - Person Data

- Group Data
- Relationships
 - Common Relationships
 - Contact Person Relationships including details and workplace address

Customer Governance (MDG-C)

Customer governance is an extension of business partner governance. It adds the SAP ERP specific parts of customer master data (the ones that are maintained in SAPGUI with transactions XD01/2/3) to business partner governance.

EhP6 Scope:

- General Data (table KNA1)

MDG 6.1 Scope:

- Company Codes (table KNB1)
- Dunning Areas (table KNB5)
- Extended Withholding Tax Types (table KNBW)
- Sales Areas (table KNVV)
- Partner Functions (table KNVP)
- Tax Indicators (table KNVI)
- Contact persons (table KNVK) via Business Partner Relationships

Supplier Governance (MDG-S)

Supplier governance is an extension of business partner governance. It adds the SAP ERP specific parts of vendor master data (the ones that are maintained in SAPGUI with transactions XK01/2/3) to business partner governance. It enables the maintenance of vendor's general data and others like company code data, purchasing organizations, and so on.

The scope of developments for business partner governance in EHP6 is as follows:

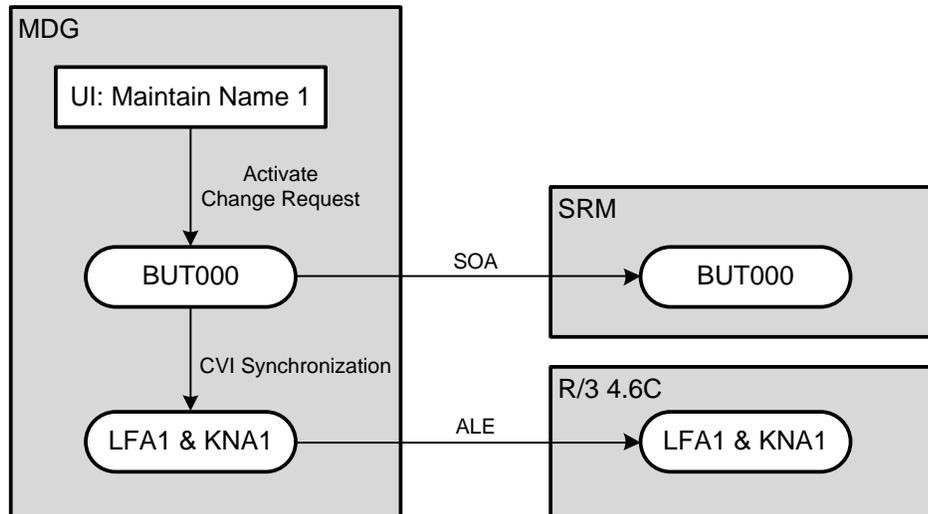
- General Data (table LFA1)
- Company Codes (table LFB1)
- Purchasing Organizations (LFM1)

The scope of developments for business partner governance in MDG 6.1 is as follows:

- Sub-Ranges (tables WYT1 & WYT1T)
- Dunning Areas (table LFB5)
- Extended Withholding Tax Types (table LFBW)
- Different Purchasing Data (table LFM2)
- Partner Functions (table WYT3)

Customer Vendor Integration (CVI)

Customer governance and supplier governance are both based on business partner governance. In MDG the business partner is the leading entity. Using the MDG UIs a business partner is always created. Depending on the Customer Vendor Integration (CVI) configuration and the entered data, a customer and/or vendor records is created in addition. Taking a detailed look at the related database tables it is obvious, that some fields are more or less identical in the different entity types. CVI is a generic tool that is able to synchronize parts of the business partner master data with the related ERP customer and vendor master data and vice versa.



The figure above shows the flow of the field **Name 1** as example. The data is maintained in the business partner part of the user interface for either customer governance or for supplier governance. As soon as the related change request is approved, data activation stores the field in the business partner table BUT000. This triggers CVI, which creates an additional entry in the related tables for customer master data (table KNA1) and vendor master data (table LFA1). One benefit of this behavior is the simplified data replication. If the newly created MDG records are sent to an SRM system, the system uses the business partner data. SRM neither knows the customer nor the vendor since this is ERP specific data. If an (older) ERP system is the target receiver, it is possible to replicate either the customer data, or the vendor data, or a combination of both, in a direct way (for example, using the common DEBMAS or CREMAS IDocs). All records in all systems have the same value for **Name 1**.

CVI synchronizes the following segments and fields:

- General Data
 - Names
 - Search Terms
 - Trading Partner
 - Business Partner Category

- ERP customer and vendor do not differentiate between organizations, persons and groups. Therefore all business partners are treated as organizations.
- Address Data
 - The business partner can handle multiple addresses. This feature is not available for ERP's customer and vendor. Therefore, only the standard address is synchronized.
- Communication Data
 - Only address dependent communication data is synchronized.
- Bank Accounts
- Industry Sectors
 - The business partner can handle multiple industry systems and sectors. This feature is not available for ERP's customer and vendor. Therefore, only the standard industry sector of the standard industry system is synchronized.
- Tax Numbers
 - The business partner stores tax numbers in a table format. ERP customer and vendor use simple fields combined with a table for European VAT Numbers. CVI takes care of the correct synchronization of the business partner table with the fields and tables for customer or vendor.

When you consider an extension of MDG, you can choose from several options to add tables and fields. Your choice strongly depends upon the following question: Which entity receives the additional information? The following scenarios are possible:

- The new field or table is available only in the business partner.
- The new field or table is available only in the customer or only in the vendor.
- The new field or table is available in both the business partner and in either the customer or in the vendor.

Each extension requires a specific sequence of steps that must be implemented correctly. Detailed how-to guides explain each extension option.

CVI is used by MDG. It is neither developed nor maintained by MDG. The synchronization happens during the activation of master data, and always in the following direction: from the business partner to the customer or the vendor. During the setup phase of Customer Governance (MDG-C) and/or Supplier Governance (MDG-S) it is possible to use the CVI Synchronization Cockpit to create business partners from existing customers or vendors.

The above mentioned use cases require a valid configuration of the CVI customizing. You can apply settings in Customizing under *Cross-Application Components* → *Master Data Synchronization* → *Customer Vendor Integration*.

MDG's core scenario of central master data governance requires that you set up Customizing to enable the synchronization direction **BP to Customer/Vendor**. This involves the following tasks:

- Defining at least one business partner role that triggers the creation of a customer/vendor.
 - It is possible to define multiple roles. If you do this, you must assign the correct role when maintaining the user interface. If you define only one role, the system uses the role automatically when it stores the business partner in the active area.
 - If you use either the **Customer Like UI** or the **Vendor Like UI**, you must define a unique role in customizing. Neither of these user interfaces can use the roles list. As a result, it is not possible to add a role manually to the current customer/vendor.
- Defining a mapping of the business partner grouping to its related customer/vendor account group.
 - The MDG UIs do not enforce the maintenance of the BP grouping. If a user does not select any BP grouping, the system chooses a default grouping based on the business partner ID in use. If a business partner ID is maintained, the default grouping for external numbering is used. If no business partner ID is maintained, the default grouping for internal numbering is used. You must correctly configure the related BP groupings when applying Customizing settings for Customer Vendor Integration.
 - The group mapping defines the key mapping implicitly since each Business Partner /Customer/Vendor group is assigned to a specific number range. Ensure that the respective number ranges match, especially if you use the same numbers for the BP and for the customer/vendor.
 - The first customer/vendor that is added to a BP is the “standard assignment”. This assignment derives its account group according to configuration specified in Customizing settings for Customer Vendor Integration. The account group cannot be changed manually in the MDG UIs. Only additional assignments allow the manual selection of the account group.
 - If you use either the Customer Like UI or the Vendor Like UI, only the following numbering combinations for Business Partners to customers/vendors are allowed:
 - BP internal and C/S internal (results in different IDs)
 - BP internal and C/S external with same number (results in identical IDs)
 - BP external and C/S external with same number (results in identical IDs)
 Other combinations would cause a deadlock in the UI.
- Making sure you check if further mapping on segment level (e.g. industry sectors) is required.

MDG processes do not require that you set up the synchronization direction **Customer/Vendor to BP**. This direction is only useful if you want to transfer existing customers or vendors to BPs. Techniques for doing this include using the MDS Load Cockpit functionality of CVI.

A detailed description of the CVI is available in the [SAP Help](#).

Multiple Assignments

The Multiple Assignment is a new sub-object used by MDG-C and MDG-S. It was introduced in release EhP6. In EhP5, MDG-S was only able to link a single business partner with a single vendor. This restriction existed due to the reuse of the Customer Vendor Integration (CVI) by MDG.

Starting with EhP6, Multiple Assignments allow you to link one or more customers or one or more vendors to a single business partner. As this is MDG-specific functionality it is only available on the MDG hub and is not available in the MDG client systems.

MDG Data Model and Access/Handler class concept

The MDG data model for business partners, customers and suppliers is model **BP**.

The entity types of the model are reuse entity types. The actual data of the model is only stored in MDG (staging) tables as long as the current record is within a governance process. As soon as the record is activated, its data is stored in existing database tables of the SAP system (for example the business partner is stored in table BUT000, the customer is stored in table KNA1, the vendor is stored in LFA1, and so on). This requires the implementation of access classes that read and store the data in the reuse area.

From EhP6, a different implementation concept for the access class is used. In EhP6, there is a single access class that distributes all calls to different, object-specific handler classes. This allows an extension of the access class simply by creating and registering a custom handler class. The access class is only responsible for controlling the calls. The actual execution of the calls, which involves, for example reading data or saving data, is implemented within the handler classes. Following the segregation of duty principle there is a handler for each object (in other words one for business partner, one for multiple assignment, one for customer, and one for supplier). This handler alone is fully responsible for the related object (in other words, the supplier handler must never change the customer object).

Software Layers

From a technical perspective, the MDG data model BP, its access classes, and its handler classes are developed in two different software layers. The general business partner components are built in the MDG foundation layer whereas the customer and supplier components belong to the MDG application layer. This is important for the further introduction of access and handler classes.

Data Derivation

The data derivation is part of the SAP handler classes. You can use data derivation to create, change, or delete data based upon either user actions in the UI or data inbound or both. An example is the automated creation of default partner functions as soon as a sales area is created in the customer UI. From the UI perspective, a derivation is triggered during the start of the UI application as well as during each UI round-trip triggered by the user. During data inbound derivations are executed, too.

The SAP handlers implement the derivation in two phases.

1. The handlers analyze the current changes applied in the User Interface or in inbound processing. The application of these changes comes via the framework to the handler class. The data is buffered within the handlers. If any actions are required due to the new data, related tasks are being buffered, too. Related coding is available in the method **BUFFER_DERIVED_DATA** of the handler classes.

2. The handlers perform the derivation according to the given tasks and buffered data. Related coding is available in the method **DERIVE_DATA** of the handler classes.

There is another special derivation option that is triggered by a change of the business partner key. In that case the MDG framework calls the handler method **DERIVE_DATA_ON_KEY_CHANGE**.

Access Classes

Access classes are required to route the calls by the MDG framework to the specific handler implementations. Since MDG-BP, MDG-C and MDG-S consist of multiple software layers, a single access class would not be sufficient for this need. Therefore, the implementation consists of three classes in a hierarchy with inheritance:

1. **CL_MDG_BS_BP_ACCESS_MASTER**

The master class is the one being called by the MDG framework. Its object instance actually depends on the current software layer it is running in. In the MDG foundation layer, it is a reference of class **CL_MDG_BS_FND_ACCESS** whereas in the MDG application layer it is a reference of class **CL_MDG_ECC_ACCESS**. The class implements the MDG framework's access interface and provides some generic functionality.

2. **CL_MDG_BS_FND_ACCESS**

This class is the MDG foundation layer implementation. It contains both generic implementations usable for all entity types of MDG-BP Customer Governance (MDG-C) and Supplier Governance (MDG-S) as well specific coding for all entity types of MDG-BP.

3. **CL_MDG_BS_ECC_ACCESS**

This class is the MDG application layer implementation. It contains only coding for all entity types of Customer Governance (MDG-C) and Supplier Governance (MDG-S).

If you want to create an extension of MDG, you do not have to enhance existing access classes respectively to create an access class of your own. We recommend that you implement a specific handler class instead.

All calls from the access to the handler classes are triggered by the access class. Therefore the access class collects a list of registered handlers consisting of both predefined SAP handlers as well as custom handlers first. Then the access class calls each handler within a loop.

It is not guaranteed that the handler class provides a fixed sequence for the calls of the access class to the different handler classes

Generic Handler Classes

The link between an access class and a handler class is the handler interface

IF_MDG_BS_BP_ACCESS_HANDLER. A handler class must implement this interface. Since major parts of the application logic needed for MDG-BP, Customer Governance (MDG-C) and Supplier Governance (MDG-S) is more or less the same for each object, two **abstract** classes (again in a hierarchy with inheritance to support the needs of the different software layers) provide this common implementation:

1. **CL_MDG_BS_FND_HANDLER**

This class is the MDG foundation layer implementation. It contains generic and reusable coding for all entity types of MDG-BP, Customer Governance (MDG-C) and Supplier Governance (MDG-S). It introduces several attributes and constants used for the data derivation (e.g. the MDG-BP specific buffer **GS_BP_EXTERN_DERIVED_DATA**).

2. **CL_MDG_BS_ECC_HANDLER**

This class is the MDG application layer implementation. It contains only reusable coding for multiple assignments, Customer Governance (MDG-C) and Supplier Governance (MDG-S). It introduces several attributes and constants used for the data derivation, such as the buffer for MDG-C and MDG-S (**GS_MLT_AS_DERIVED_DATA**) and task buffers (**GV_BP_ROOT_TASK** or **GT_MLTAS_TASK**).

The above-mentioned classes are abstract. A few public classes that from the abstract classes and provide further object-specific logic. The classes follow the segregation of duty principle. A single handler is responsible for a single object only. The following chapters introduce the classes in detail.

If you want to extend MDG by creating your own handler classes, there are several possibilities. A general recommendation for the creation of a custom handler class is that your new class inherits from either the abstract foundation handler or from the application handler. Deciding which parent to choose from the available SAP handlers depends on the actual use case of the extension. In some cases, it might even make sense to choose one of the SAP predefined object handlers as a parent.

Handler Class for Business Partner Governance (MDG-BP)

The MDG-BP specific handler class is **CL_MDG_BS_BP_HANDLER**. It inherits from class **CL_MDG_BS_FND_HANDLER**. It provides the business partner-specific implementation of the handler interface.

This class is a valid parent for a custom handler class if the use case is a modification or enhancement of the existing application logic of the business partner.

Handler Class for Multiple Assignments

The multiple assignment specific handler class is **CL_MDG_BS_MLT_ASSGNMNT_HANDLER**. It inherits from class **CL_MDG_BS_ECC_HANDLER**. It provides the multiple assignment-specific implementation of the handler interface.

This class is valid parent for a custom handler class if the use case is a modification or enhancement of the existing application logic of the multiple assignments.

Handler Class for Customer Governance (MDG-C)

The MDG-C specific handler class is **CL_MDG_BS_CUST_HANDLER**. It inherits from class **CL_MDG_BS_ECC_HANDLER**. It provides the customer specific implementation of the handler interface.

This class is valid parent for a custom handler class if the use case is a modification or enhancement of the existing application logic of the customer.

Handler Class for Supplier Governance (MDG-S)

The MDGS specific handler class is `CL_MDG_BS_SUPPL_HANDLER`. It inherits from class `CL_MDG_BS_ECC_HANDLER`. It provides the supplier (vendor) specific implementation of the handler interface.

This class is valid parent for a custom handler class if the use case is a modification or enhancement of the existing application logic of the supplier (vendor).

Modeling and Implementation for the Generic Interaction Layer (genIL)

Since the MDG BP UIs are built using Floorplan Manager (FPM) and its Business Object Layer (BOL) / Generic Integration Layer (genIL) integration, the genIL model builds the foundation of the UI.

The genIL layer consists of a genIL model and one or more genIL implementation classes for the specific model. genIL models are defined and maintained in the SAP backend with transaction `GENIL_MODEL_BROWSER`. genIL implementation classes are built within the common transactions `SE24` and/or `SE80`.

The genIL model of MDG-BP, MDG-C and Supplier Governance (MDG-S) reflects the introduced separation of objects, too. The basis is the business partner genIL model component `BUPA` which is extended by customer and supplier in the genIL model enhancement `BUPA_CUSP`.

A genIL model basically consists of objects and relations.

- Objects consist of attributes. Each attribute reflects a usable field for the user interface.
- Relations connect one object to another. They define the cardinality of objects in a relation, too. Relations are reflected in the user interface by the wires (connections) from one UIBB to another. It is mandatory that the UIBB hierarchy in the overview page is consistent to the genIL object hierarchy as defined by the relations.

The important parts of an object are the key structure and the attribute structure. Both must be existing ABAP DDIC structures. The key structure is used internally by genIL. **The sub-object's key structure must always contain the key fields of its parent.** The attribute structure is used by FPM during the UI creation. It defines the field catalogue that is available for creating a list or form UIBB. **If a key field shall be visible respectively maintainable in the UI, it must be part of the attribute structure, too.**

Each genIL model component respectively enhancement requires exactly one implementation class. With regards to the complexity of the MDG-BP, MDG-C and Supplier Governance (MDG-S) solution this might be considered as a "limitation" since the MDG solutions would need implementation classes for each specific object. To overcome this "limitation" the MDG-BP, MDG-C and MDG-S genIL implementation consists of a class hierarchy where each class inherits from its parent class:

1. **CL_MDG_BS_GENIL**: the generic genIL implementation for all MDG solutions
2. **CL_BS_GENIL_BUPA**: the genIL implementation for MDGBP
3. **CL_BS_GENIL_MLT_ASSIGNMENTS**: the genIL implementation for multiple assignments
4. **CL_BS_GENIL_SUPPLIER**: the genIL implementation for MDGS
5. **CL_BS_GENIL_CUSTOMER**: the genIL implementation for all MDGC

The last class CL_BS_GENIL_CUSTOMER is the one that is defined in the genIL model enhancement BUPA_CUSP. The class hierarchy ensures that always all classes are executed during design and runtime of the UI. It is absolutely mandatory that each custom enhancement of the genIL model which needs to use an own implementation class inherits from CL_BS_GENIL_CUSTOMER. Otherwise it might come to unforeseen errors during design and runtime of the UI.

The generic implementation in class CL_MDG_BS_GENIL contains a type based mapping from the UI data structures into the MDG entity data structures and vice versa. This mapping works fine if the used data type of the source and target field is identical. If this is not the case, or if multiple fields are using the same data type, the generic mapping fails which might lead to disappearing values in the UI. To resolve this issue, it is possible to re-define one of the following methods:

- IF_BS_TYPECASTED_MAP_ASSISTANT~TARGET_FIELD_NAME
 - The method allows defining a strict field mapping based upon the source structure and field name to the target structure and field name.
- IF_BS_TYPECASTED_MAP_ASSISTANT~TYPE_ALTERNATIVE
 - The method allows defining an alternative data type for the generic type based mapping based upon the source structure and field type to the target structure and field type.

“Example” implementations are pre-defined by SAP in the above mentioned object-specific classes (items 2 to 4).

MDG-BP

The following genIL objects are modeled.

1. BP_Root representing the root business partner object.
2. BP_Group being a sub-object of BP_Root representing the Group specific business partner data.
3. BP_Organization being a sub-object of BP_Root representing the Organization specific business partner data.
4. BP_Person being a sub-object of BP_Root representing the Person specific business partner data.
5. ADDR_Root representing the address.
6. ADDR_CommPhone representing the phone communication data of an address.
7. BP_Relation representing the business partner relationships.
8. BP_Contact_Person being a sub-object of BP_Relation representing the special contact person relation relationship.
9. BP_Address being a sub-object of ADDR_Root representing the business partner address.

10. BP_AddressUsage representing the usage of an address.
11. BP_AddressVersion_Organization representing the international address versions for business partners of type Organization.
12. BP_AddressVersion_Person representing the international address versions for business partners of type Person.
13. BP_BankAccount representing the bank account resp. bank details.
14. BP_CommEmail representing the email address.
15. BP_CommFax representing the fax data.
16. BP_CommMobile representing the mobile phone data.
17. BP_CommPhone representing the common phone data.
18. BP_CommURI representing the web page data.
19. BP_IdentificationNumber representing the identification number data.
20. BP_Industry representing the industry sectors.
21. BP_PersonVersion representing person data for international address versions.
22. BP_Role representing business partner roles.
23. BP_TaxNumbers representing tax information data.
24. BP_WorkplaceAddress representing the workplace address of a contact person.
25. BP_CommEmail representing the workplace email address of a contact person.
26. BP_CommFax representing the workplace fax data of a contact person.
27. BP_CommMobile representing the workplace mobile phone data of a contact person.
28. BP_CommPhone representing the workplace common phone data of a contact person.
29. BP_CommURI representing the workplace web page data of a contact person.
30. BP_QueryRoot representing a static query object usable for searching business partners.
31. BP_DynamicQueryMDGProviderRoot representing a dynamic query object usable for searching business partners.

Multiple Assignments

The following genIL objects are modeled.

1. BP_MultipleAssignment representing the multiple assignments.

The following genIL relations are modeled. All relations link the business partner root object (BP_Root) with the multiple assignment object (BP_MultipleAssignment).

Name	Description
BP_MultipleAssignmentRel	This is a generic 0:n relation. It is not used by the SAP standard. It prepares a custom multiple assignment category.
CU_MultipleAssignmentsRel	This is a customer specific 0:n relation. It is used by MDG-C. Using filter mechanisms it ensures that during MDG-C usage only customers are visible in the UI.

Name	Description
CU_StandardAssignmentRel	This is a customer specific 0:1 relation. It is used by MDG-C. Using filter mechanisms it ensures that during MDG-C usage only the standard assignment for customers is visible in the UI.
SP_MultipleAssignmentsRel	This is a supplier specific 0:n relation. It is used by Supplier Governance (MDG-S). Using filter mechanisms it ensures that during MDG-S usage only vendors are visible in the UI.
SP_StandardAssignmentRel	This is a supplier specific 0:1 relation. It is used by Supplier Governance (MDG-S). Using filter mechanisms it ensures that during MDG-S usage only the standard assignment for vendors is visible in the UI.

All genIL specific ABAP objects referring to multiple assignments are available in package **MDG_BS_ECC_BP_MLT_ASSGMNT**. The implementation class is **CL_BS_GENIL_MLT_ASSIGNMENTS**.

The only key structure is **BSS_BPIL_MLT_ASSGMNT_KEY**. It is used by several objects. It is the key for multiple assignments (BP_MultipleAssignment) as well as customer and vendor general data (CU_GeneralData and SP_GeneralData). The latter one is possible due to the fact that the multiple assignments key already uniquely identifies a customer and vendor general data object.

The only attribute structure is **BSS_BPIL_MLT_ASSGMNT_ATTR** for object BP_MultipleAssignment.

MDG-C

The following genIL objects are modeled.

1. CU_GeneralData representing the general data.
2. CU_CompanyCode representing the company codes.
3. CU_DunningArea representing the dunning areas.
4. CU_WithholdingTaxType representing the extended withholding tax types.
5. CU_SalesArea representing the sales areas.
6. CU_PartnerFunction representing the partner functions.
7. CU_TaxIndicator representing the tax indicators.

The following genIL relations are modeled.

Name	Description
CU_AssignedCustomerRel	This is a customer specific 0:1 relation. It links the customer multiple assignment (BP_MultipleAssignment) with its general data (CU_GeneralData).

Name	Description
CU_AssignedCompanyCodesRel	This is a customer specific 0:n relation. It links the customer multiple assignment (BP_MultipleAssignment) with its company codes (CU_CompanyCode).
CU_CompDunningAreasRel	This is a customer specific 0:n relation. It links the customer company code (CU_CompanyCode) with its dunning areas (CU_DunningArea).
CU_CompWithholdingTaxesRel	This is a customer specific 0:n relation. It links the customer company code (CU_CompanyCode) with its extended withholding tax types (CU_WithholdingTaxType).
CU_AssignedSalesAreasRel	This is a customer specific 0:n relation. It links the customer multiple assignment (BP_MultipleAssignment) with its sales areas (CU_SalesArea).
CU_SalesPartnerFunctionsRel	This is a customer specific 0:n relation. It links the customer sales area (CU_SalesArea) with its partner functions (CU_PartnerFunction).
CU_AssignedTaxIndicatorsRel	This is a customer specific 0:n relation. It links the customer multiple assignment (BP_MultipleAssignment) with its tax indicators (CU_TaxIndicator). This relation is used for displaying all tax indicators of a customer.
CU_SalesTaxIndicatorsRel	This is a customer specific 0:n relation. It links the customer sales area (CU_SalesArea) with its tax indicators (CU_TaxIndicator). This relation is used for displaying the tax indicators per sales area in the UI.

All genIL specific ABAP objects referring to customers are available in package

MDG_BS_ECC_CUSTOMER_GENIL. The implementation class is **CL_BS_GENIL_CUSTOMER**.

Key Structure	Description
BSS_CUIL_COMPANY_CODE_KEY	Used for object CU_CompanyCode. It includes the multiple assignment keys and adds the company code key.
BSS_CUIL_DUNNING_KEY	Used for object CU_DunningArea. It includes the company code keys and adds the dunning area key.
BSS_CUIL_WHTAX_KEY	Used for object CU_WithholdingTaxType. It includes the company code keys and adds the withholding tax keys. The country is required due to MDG modeling.
BSS_CUIL_SALES_AREA_KEY	Used for object CU_SalesArea. It includes the multiple assignment keys and adds the sales areas keys.

Key Structure	Description
BSS_CUIL_FUNCTION_KEY	Used for object CU_PartnerFunction. It includes the sales area keys and adds the partner function keys.
BSS_CUIL_TAXINDICATOR_KEY	Used for object CU_TaxIndicator. It includes the multiple assignment keys and adds the tax indicator keys. Although tax indicators are only calculated if a sales area is maintained, they do not depend on the sales area keys.

Attribute Structure	Description
BSS_CUIL_GENERAL	Used for object CU_GeneralData. The customer ID is treated as a common attribute.
BSS_CUIL_COMPANY_CODE	Used for object CU_CompanyCode. It includes the company code key as attribute to enable its maintenance in the UI.
BSS_CUIL_DUNNING	Used for object CU_DunningArea. It includes the company code key as attribute to enable its maintenance in the UI.
BSS_CUIL_WHTAX	Used for object CU_WithholdingTaxType. It includes only the withholding tax type as attribute to enable its maintenance in the UI. The country is not displayed.
BSS_CUIL_SALES_AREA	Used for object CU_SalesArea. It includes the sales area keys as attributes to enable their maintenance in the UI.
BSS_CUIL_FUNCTION	Used for object CU_PartnerFunction. It includes the partner function keys as attributes to enable their maintenance in the UI. The partner counter is not displayed in the UI.
BSS_CUIL_TAXINDICATORS	Used for object CU_TaxIndicator. It includes the tax indicator keys as attributes to enable their display in the UI.

Supplier Governance (MDG-S)

The following genIL objects are modeled.

1. SP_GeneralData representing the general data.
2. SP_CompanyCode representing the company codes.
3. SP_DunningArea representing the dunning areas.
4. SP_WithholdingTaxType representing the extended withholding tax types.
5. SP_PurchasingOrg representing the purchasing organization.
6. SP_PurchasingOrg2 representing the different purchasing data.

7. SP_PartnerFunction representing the partner functions.
8. SP_Subrange representing the vendor sub-ranges.

The following genIL relations are modeled.

Name	Description
SP_AssignedSupplierRel	This is a supplier specific 0:1 relation. It links the supplier multiple assignment (BP_MultipleAssignment) with its general data (SP_GeneralData).
SP_AssignedCompanyCodesRel	This is a supplier specific 0:n relation. It links the supplier multiple assignment (BP_MultipleAssignment) with its company codes (SP_CompanyCode).
SP_SingleCompanyCodeRel	This is a supplier specific 0:1 relation. It links the supplier multiple assignment (BP_MultipleAssignment) with a single company code (SP_CompanyCode). It is used by the lean creation UI.
SP_CompDunningAreasRel	This is a supplier specific 0:n relation. It links the supplier company code (SP_CompanyCode) with its dunning areas (SP_DunningArea).
SP_CompWithholdingTaxesRel	This is a supplier specific 0:n relation. It links the supplier company code (SP_CompanyCode) with its extended withholding tax types (SP_WithholdingTaxType).
SP_AssignedPurchasingOrgsRel	This is a supplier specific 0:n relation. It links the supplier multiple assignment (BP_MultipleAssignment) with its purchasing organizations (SP_PurchasingOrg).
SP_SinglePurchasingOrgRel	This is a supplier specific 0:1 relation. It links the supplier multiple assignment (BP_MultipleAssignment) with a single purchasing organization (SP_PurchasingOrg). It is used by the lean creation UI.
SP_PurchOrgPurchasingOrg2Rel	This is a supplier specific 0:n relation. It links the supplier purchasing organization (SP_PurchasingOrg) with its different purchasing data (SP_PurchasingOrg2).
SP_PurchOrgPartnerFunctionsRel	This is a supplier specific 0:n relation. It links the supplier purchasing organization (SP_PurchasingOrg) with its partner functions (SP_PartnerFunction).
SP_AssignedSubrangesRel	This is a supplier specific 0:n relation. It links the supplier multiple assignment (BP_MultipleAssignment) with its sub-ranges (SP_Subrange).

All genIL specific ABAP objects referring to suppliers are available in package

MDG_BS_ECC_SUPPLIER_GENIL. The implementation class is **CL_BS_GENIL_SUPPLIER**.

Key Structure	Description
BSS_SPIL_COMPANY_CODE_KEY	Used for object SP_CompanyCode. It includes the multiple assignment keys and adds the company code key.
BSS_SPIL_DUNNING_KEY	Used for object SP_DunningArea. It includes the company code keys and adds the dunning area key.
BSS_SPIL_WHTAX_KEY	Used for object SP_WithholdingTaxType. It includes the company code keys and adds the withholding tax keys. The country is required due to MDG modeling.
BSS_SPIL_PURCHASING_ORG_KEY	Used for object SP_PurchasingOrg. It includes the multiple assignment keys and adds the purchasing organization key.
BSS_SPIL_PURCHASING_ORG2_KEY	Used for object SP_PurchasingOrg2. It includes the purchasing organization key and adds the different purchasing data keys.
BSS_SPIL_FUNCTION_KEY	Used for object SP_PartnerFunction. It includes the purchasing organization key and adds the partner function keys.
BSS_SPIL_SUBRANGE_KEY	Used for object SP_Subrange. It includes the multiple assignment keys and adds the sub-range keys.

Attribute Structure	Description
BSS_SPIL_GENERAL	Used for object SP_GeneralData. The vendor ID is treated as a common attribute.
BSS_SPIL_COMPANY_CODE	Used for object SP_CompanyCode. It includes the company code key as attribute to enable its maintenance in the UI.
BSS_SPIL_DUNNING	Used for object SP_DunningArea. It includes the company code key as attribute to enable its maintenance in the UI.
BSS_SPIL_WHTAX	Used for object SP_WithholdingTaxType. It includes only the withholding tax type as attribute to enable its maintenance in the UI. The country is not displayed.
BSS_SPIL_PURCHASING_ORG	Used for object SP_PurchasingOrg. It includes the purchasing organization key as attribute to enable its maintenance in the UI.

Attribute Structure	Description
BSS_SPIL_PURCHASING_ORG2	Used for object SP_PurchasingOrg2. It includes the different purchasing data keys as attributes to enable their maintenance in the UI.
BSS_SPIL_FUNCTION	Used for object SP_PartnerFunction. It includes the partner function keys as attributes to enable their maintenance in the UI. The partner counter is not displayed in the UI.
BSS_SPIL_SUBRANGE	Used for object SP_Subrange. It includes the sub-range keys as attributes to enable their display in the UI.

User Interfaces

The MDG UIs are built with the floor plan manager (FPM) using the Business Object Layer (BOL) / Generic Integration Layer (genIL) technology. The UI uses a specific genIL model that differs from the actual MDG BP model used for data governance. Some advantages to be mentioned are:

- Loose coupling of the UIs to the MDG specific processes
- High flexibility for the creation of the UIs. The huge amount of fields to be displayed is split into small UI Building Blocks (UIBBs). UIBBs support lists, forms and special kinds like pop-ups, search input and search results.
- Possibility to create object specific UIs to create a common look and feel and/or a similarity of the MDG UI compared to the SAPGUI maintenance transactions
- Reuse of existing tables, structures and fields (including naming) during the UI creation

General information about FPM and its functionality is available in the [FPM Cookbook on SDN](#).

Context Based Adaptations

A context based adaptation (CBA) is a FPM concept that allows changing the UI in a flexible way based upon given values (e.g. application parameters, user input, and others). A CBA consists of an **Adaptation Schema** that consists of one or more **Adaptation Dimensions**. Using both it is possible to create various adaptations of the UI (e.g. a different layout of an overview page (OVP), or additional/removed row actions within a list UIBB, and so on). It is possible to combine several dimensions to create a very specific adaptation.

The CBA concept is based on the common FPM event handling. It is possible to trigger one or more CBA events that are handled by FPM's event loop processing.

Refer to chapter **Context Based Adaptations (CBA)** in the [FPM Cookbook on SDN](#) for more details.

The layout of an OVP can only be changed with a CBA during the startup of the application. It is not possible to change the OVP (e.g. the sequence of UIBBs) using a CBA during UI round-trips. CBAs can only change the layout of single UIBBs for each round-trip.

FPM's event handling concept allows triggering multiple CBA events during one UI round-trip. But FPM does not cumulate the dimension information given within each event. Consider a scenario with the following sequence of events:

1. A first event triggers a CBA for dimension ACTION (for example by defining the value DELETE to trigger the Mark for Deletion UI),
2. A second CBA event contains only some information for a different dimension without the dimension ACTION.

In this scenario, the CBA for the ACTION will not be NOT processed since its value is reset to initial

Still, there is a possibility to trigger complex CBAs described in the related use case Trigger my own CBA Event.

The SAP delivered UIs support and use CBAs. There is a predefined schema **BP_ADAPTS** having several dimensions that are actively used within the UIs. The actual UI that is being displayed to a user in the web browser is determined from various components of the UI configuration:

- Personalization
- Enhancements
- Context Based Adaptations
- Base Configuration

The general rule is that the personalization is the strongest component. This is best explained with an example.

The base configuration defines the overview page as a list of UIBBs. Since a user does not want to scroll, he or she creates a personalization of the page introducing a stacking of the UIBBs in tab-strips. A UI designer decides to create a context based adaptation that sets a single UIBB to "hidden and excluded from event loop". All users not having a personalization will not see this UIBB anymore. The user with the personalization set is unaffected by this change. This is because the UIBBs that are hidden and excluded from event loop still belong to the OVP. They can be added to the OVP using personalization. Since the user has created a personalization that shows the UIBB (the personalization was created before the CBA), the UIBB is still visible. To exclude the UIBB you must either reset personalization or delete the UIBB in the CBA.

MDG-BP

[Overview](#)

[ABAP Object List](#)

Multiple Assignments

[Overview](#)

The multiple assignment UI consists of re-usable components only. There is no specific OVP. There are several lists available for the different assignment categories of business partner, customer and supplier. This simplifies the creation of object specific OVPs. It allows the usage of several multiple assignment lists on a single OVP. Additionally there is a single form for maintaining the multiple assignment data.

All UI specific ABAP objects for the multiple assignments are available in package **MDG_BS_ECC_BP_MLT_ASSGMNT**.

[ABAP Object List](#)

The genIL object defining the field catalogue is **BP_MultipleAssignment**.

Class **CL_BS_BP_GUIBB_MLT_ASSIGNMENTS** is the responsible feeder class for the below mentioned list UIBBs.

List UIBB	Description
BS_BP_MLT_ASSIGNMENTS	A common list for assignments that is not used by the SAP delivered UI configurations. The list can be used to add custom assignment categories to the UI.
BS_CU_MLT_ASSIGNMENTS	The customer specific list that links the business partner with one or more SAP ERP customers.
BS_SP_MLT_ASSIGNMENTS	The supplier specific list that links the business partner with one or more SAP ERP vendors.

Class **CL_BS_BP_GUIBB_MLT_ASSIGNMENT** is the responsible feeder class for the form UIBB **BS_BP_MLT_ASSIGNMENT**. This is a common form for assignments that is not used by the SAP delivered UI configurations. It can be used to maintain custom assignment categories in details.

Customer Governance (MDG-C)

Overview

The Customer Governance (MDG-S) UI provides several independent UI applications for maintaining supplier master data. All applications re-use the business partner UI and either change or extend it specifically for the supplier.

User Interface	Description
BS_OVP_CU	<p>Main UI for customer governance.</p> <p>Reuses the overview page for the business partner UI.</p> <p>Extends this page with the customer multiple assignments list. This list is the starting point for the maintenance of ERP customer master data.</p> <p>The UI consists of several edit pages with forms and list User Interface Building Blocks (UIBB).</p>
BS_OVP_CU_VL	<p>Referred to as the customer-like user interface.</p> <p>Does not use multiple assignments.</p> <p>Layout is synchronized, as far as possible, with the SAP GUI transactions used to maintain customers.</p>

All UI specific ABAP objects for the customer are available in package **MDG_BS_ECC_CUSTOMER_BOLUI**.

ABAP Object List: Generic Components

The main application configuration is **BS_OVP_CU**. Starting this application the overview page **BS_CU_OVP** is called with the application specific communicator setting as defined in **BS_OVP_CU**. Any customer UI specific messages are stored within message class **MDG_BS_CUST_BOLUI**.

ABAP Object List: Enterprise Search

The enterprise search template includes customer attributes like the customer ID. To reflect this within the search UIBB, a customer specific search UIBB **BS_CU_DQUERY_ES** is provided. Its feeder class is **CL_BS_CU_GUIBB_DQUERY**.

ABAP Object List: General Data

The genIL object defining the field catalogue is **CU_GeneralData**.

Class **CL_BS_CU_GUIBB_GENERAL_DATA** is the responsible feeder class for the below mentioned form UIBBs. The class is a join feeder since it handles both the multiple assignments and customer general data attributes.

List UIBB	Description
BS_CU_GEN_ADDITIONAL	A form for the "additional data" segment of the customer's general data.
BS_CU_GEN_ALL	A form consisting of all segments of the customer's general data. This form is currently not used by the SAP delivered UI configurations. It could be used to replace the multiple single form UIBBs, e.g. in case of performance issues.
BS_CU_GEN_BLOCK_DEL	A form for the "blocking and deletion flag" segments of the customer's general data.
BS_CU_GEN_CONTROL	A form for the "control data" segment of the customer's general data and the multiple assignment attributes.
BS_CU_GEN_EXPORT	A form for the "export data" segment of the customer's general data.
BS_CU_GEN_MARKETING	A form for the "marketing" segment of the customer's general data.
BS_CU_GEN_PAYMENT	A form for the "payment transactions" segment of the customer's general data.
BS_CU_GEN_TAXINFORMATION	A form for the "tax information" segment of the customer's general data.

ABAP Object List: Company Codes

The genIL object defining the field catalogue is **CU_CompanyCode**.

Class **CL_BS_CU_COMPANY_CODES** is the responsible feeder class for the list UIBB **BS_CU_COMPANY_CODES**. In the standard UI the list is part of the customer's general data edit page. It shows the company code key and description only. It is read-only since the actual data is maintained by forms on a specific edit page.

Class **CL_BS_CU_COMPANY_CODE** is the responsible feeder class for the below mentioned form UIBBs.

Form UIBB	Description
BS_CU_COMPANY_CODE	A form for displaying the key fields of customer's company code.

Form UIBB	Description
BS_CU_COMPANY_CODE_ALL	A form consisting of all segments of the customer's company code. This form is currently not used by the SAP delivered UI configurations. It could be used to replace the multiple single form UIBBs, e.g. in case of performance issues.
BS_CU_COMPANY_CODE_BLOCK_DEL	A form for the "blocking and deletion flag" segments of the customer's company code.
BS_CU_COMPANY_CODE_ACCOUNTING	A form for displaying the "accounting information" segment of the customer's company code.
BS_CU_COMPANY_CODE_INTEREST	A form for displaying the "interest calculation data" segment of the customer's company code.
BS_CU_COMPANY_CODE_REFERENCE	A form for displaying the "reference data" segment of the customer's company code.
BS_CU_COMPANY_CODE_PAYM_DATA	A form for displaying the "payment data" segment of the customer's company code.
BS_CU_COMPANY_CODE_AUTO_PAY	A form for displaying the "automatic payment transactions" segment of the customer's company code.
BS_CU_COMPANY_CODE_PAYM_ADVICE	A form for displaying the "payment advice notes" segment of the customer's company code.
BS_CU_COMPANY_CODE_EXP_INSURANCE	A form for displaying the "export credit insurance" segment of the customer's company code.
BS_CU_COMPANY_CODE_DUNNING	A form for displaying the "dunning data" segment of the customer's company code.

ABAP Object List: Dunning Areas

The genIL object defining the field catalogue is **CU_DunningArea**.

Class **CL_BS_CU_DUNNING_AREAS** is the responsible feeder class for the list UIBB **BS_CU_DUNNING_AREAS**. In the standard UI the list is part of the customer's company code edit page. It shows the dunning area key and description only. It is read-only since the actual data is maintained by a form on a specific edit page.

Class **CL_BS_CU_DUNNING_AREA** is the responsible feeder class for the form UIBB **BS_CU_DUNNING_AREA**.

ABAP Object List: Extended Withholding Tax Types

The genIL object defining the field catalogue is **CU_WithholdingTaxType**.

Class **CL_BS_CU_WHTAXES** is the responsible feeder class for the list UIBB **BS_CU_WITHHOLDING_TAXES**. In the standard UI the list is part of the customer's company code edit page. It shows the extended withholding tax type key and description only. It is read-only since the actual data is maintained by a form on a specific edit page. Maintenance in general is only allowed if the parent company code uses the extended withholding tax functionality.

Class **CL_BS_CU_WHTAX** is the responsible feeder class for the form UIBB **BS_CU_WITHHOLDING_TAX**.

ABAP Object List: Sales Areas

The genIL object defining the field catalogue is **CU_SalesArea**.

Class **CL_BS_CU_SALES_AREAS** is the responsible feeder class for the list UIBB **BS_CU_SALES_AREAS**. In the standard UI the list is part of the customer's general data edit page. It shows the sales area keys and description only. It is read-only since the actual data is maintained by forms on a specific edit page.

Class **CL_BS_CU_SALES_AREA** is the responsible feeder class for the below mentioned form UIBBs.

Form UIBB	Description
BS_CU_SALES_AREA	A form for displaying the key fields of the customer's sales area.
BS_CU_SALES_AREA_ALL	A form consisting of all segments of the customer's sales area. This form is currently not used by the SAP delivered UI configurations. It could be used to replace the multiple single form UIBBs, e.g. in case of performance issues.
BS_CU_SALES_AREA_BLOCK_DEL	A form for displaying the blocking and deletion flags of the customer's sales area. This UIBB is visible only in blocking mode.
BS_CU_SALES_AREA_ORDER	A form for displaying the "sales orders" segment of the customer's sales area.
BS_CU_SALES_AREA_PRICING_STATS	A form for displaying the "pricing and statistics" segment of the customer's sales area.
BS_CU_SALES_AREA_AGENCY_BUSINESS	A form for displaying the "agency business" segment of the customer's sales area.
BS_CU_SALES_AREA_SHIPPING	A form for displaying the "shipping" segment of the customer's sales area.

Form UIBB	Description
BS_CU_SALES_AREA_PART_DELIVER	A form for displaying the “partial deliveries” segment of the customer’s sales area.
BS_CU_SALES_AREA_BILLING_DOC	A form for displaying the “billing documents” segment of the customer’s sales area.
BS_CU_SALES_AREA_DELIVER_PAYM	A form for displaying the “delivery and payment terms” segment of the customer’s sales area.
BS_CU_SALES_AREA_ACCOUTING	A form for displaying the “accounting” segment of the customer’s sales area.
BS_CU_SALES_AREA_CUSTOMER_GROUPS	A form for displaying the “customer groups” segment of the customer’s sales area.

ABAP Object List: Partner Functions

The genIL object defining the field catalogue is **CU_PartnerFunction**.

Class **CL_BS_CU_FUNCTIONS** is the responsible feeder class for the list UIBB **BS_CU_FUNCTIONS**. In the standard UI the list is part of the customer’s sales area edit page. It shows the complete partner function data. The data is maintained within the list. There’s no specific edit page.

ABAP Object List: Tax Indicators

The genIL object defining the field catalogue is **CU_TaxIndicator**.

Class **CL_BS_CU_TAX_INDICATORS** is the responsible feeder class for two list UIBBs. List **BS_CU_TAX_INDICATORS** is part of the customer’s general data edit page. It shows all tax indicators of the customer.

BS_CU_SALES_TAX_INDICATORS is part of the customer’s sales area edit page. It shows all tax indicators for the current sales area of the customer.

For both the data is maintained within the list. There’s no specific edit page.

ABAP Object List: Customer-like UI

The Customer-like UI re-uses most of the Customer Governance (MDG-C) customer UI forms and lists. Still, it is necessary to provide additional UIBBs where the common ones cannot be used. The following are UIBBs that replace the common ones:

- Form UIBB **BS_CU_CL_GEN_ADMIN** is used additionally for administrative data of the Customer Like UI. This form has a specific feeder class **CL_BS_CU_GUIBB_Customer**.

- Composite UIBB **BS_CU_CL_DETAILS_CU** is used instead of the common composite UIBB of the customer.
- Form UIBB **BS_CU_CL_GEN_BLOCK** is used instead of the common form UIBB for the “blocking flags” of the customer.
- Form UIBB **BS_CU_CL_GEN_CONTROL** is used instead of the common form UIBB for the “control data” segment of the customer.
- Form UIBB **BS_CU_CL_GEN_DELETE** is used instead of the common form UIBB for the “deletion flags” of the customer.
- List UIBB **BS_CU_CL_QUERY_RESULT** is used instead of the common list UIBB for customer search results.
- Form UIBB **BS_CU_CL_RELATION** is used instead of the common form UIBB for business partner relationships.
- List UIBB **BS_CU_CL_RELATIONS** is used instead of the common list UIBB for business partner relationships.

ABAP Object List: Lean Vendor Creation UI

The Lean Vendor Creation UI re-uses most of the Supplier Governance (MDG-S) supplier UI forms and lists (the major part of the UIBBs are hidden). Still, it is necessary to provide additional UIBBs where the common ones cannot be used. The following are UIBBs that replace the common ones:

- Composite UIBB **BS_SP_LVC_DETAILS_CU** is used instead of the common composite UIBB of the supplier.
- Form UIBB **BS_SP_LVC_ORG_UNITS** is an additional form UIBB to allow the request for a company code and purchasing organization. It consists of the key fields only. This form has a specific feeder class **CL_BS_SP_GUIBB_LVC_ORG_UNITS**.
- List UIBB **BS_SP_LVC_QUERY_RESULT** is used instead of the common list UIBB for supplier search results.

Supplier Governance (MDG-S)

Overview

The MDGS UI provides an independent UI application for maintaining customer master data. Therefore it re-uses the overview page of the business partner UI. It extends it with the supplier multiple assignments list. This list is the starting point for the maintenance of ERP supplier master data. The UI consists of several edit pages with form and list UIBBs.

All UI specific ABAP objects for the supplier are available in package **MDG_BS_ECC_SUPPLIER_BOLUI**.

The Supplier Governance (MDG-S) UI provides several independent UI applications for maintaining supplier master data. All applications re-use the business partner UI and either change or extend it specifically for the supplier.

User Interface	Description
BS_OVP_SP	<p>Main UI for supplier governance.</p> <p>Reuses the overview page for the business partner UI.</p> <p>Extends this page with the vendor multiple assignments list. This list is the starting point for the maintenance of ERP vendor master data.</p> <p>The UI consists of several edit pages with forms and list User Interface Building Blocks (UIBB).</p>
BS_OVP_SP_VL	<p>Referred to as the vendor-like user interface.</p> <p>Does not use multiple assignments.</p> <p>Layout is synchronized, as far as possible, with the SAP GUI transactions used to maintain vendors.</p>
BS_OVP_SP_LVC	<p>Referred to as the Lean Vendor Creation UI.</p> <p>Provides a single request form to trigger the creation of a supplier with ERP vendor master data.</p>

The main Supplier Governance (MDG-S) UI **BS_OVP_SP** for suppliers it re-uses the overview page of the business partner UI. It extends it with the vendor multiple assignments list. This list is the starting point for the maintenance of ERP vendor master data. The UI consists of several edit pages with form and list UIBBs.

The Supplier Governance (MDG-S) UI **BS_OVP_SP_VL** (so called **Vendor Like UI**) does not use multiple assignments. The layout of this UI is synchronized (as far as possible) with the SAGUI transactions for ERP vendor maintenance.

The Supplier Governance (MDG-S) UI **BS_OVP_SP_LVC** (so called **Lean Vendor Creation UI**) provides a single request form to trigger the creation of a supplier with ERP vendor master data.

All UI specific ABAP objects for the supplier are available in package **MDG_BS_ECC_SUPPLIER_BOLUI**.

ABAP Object List: Generic Components

There are three application configurations to start the different UIs.

BS_OVP_SP starts the overview page **BS_SP_OVP** with the application specific communicator setting as defined in **BS_OVP_SP**.

BS_OVP_SP_VL starts the overview page **BS_SP_VL_OVP** with the application specific communicator setting as defined in **BS_OVP_SP_VL**.

BS_OVP_SP_LVC starts the overview page **BS_SP_LVC_OVP** with the application specific communicator setting as defined in **BS_OVP_SP_LVC**.

Any supplier UI specific messages are stored within message class **MDG_BS_SUPP_BOLUI**.

ABAP Object List: Enterprise Search

The enterprise search template includes vendor attributes like the vendor ID, company codes, and so on. To reflect this within the search UIBB, a vendor specific search UIBB **BS_SP_DQUERY_ES** is provided. Its feeder class is **CL_BS_SP_GUIBB_DQUERY**.

ABAP Object List: General Data

The genIL object defining the field catalogue is **SP_GeneralData**.

Class **CL_BS_SP_GUIBB_GENERAL_DATA** is the responsible feeder class for the below mentioned form UIBBs. The class is a join feeder since it handles both the multiple assignments and vendor general data attributes.

List UIBB	Description
BS_SP_GEN_CONTROL	A form UIBB for displaying the segment “control data” of the vendor's general data and multiple assignments attributes.
BS_SP_GEN_ALL	A form consisting of all segments of the vendor's general data. This form is currently not used by the SAP delivered UI configurations. It could be used to replace the multiple single form UIBBs, e.g. in case of performance issues.
BS_SP_GEN_REFERENCE	A form UIBB for displaying the segment “reference data” of the vendor's general data.
BS_SP_GEN_TAXINFORMATION	A form UIBB for displaying the segment “tax information” of the vendor's general data.
BS_SP_GEN_PAYMENT	A form UIBB for displaying the segment “payment transaction” data of the vendor's general data.

List UIBB	Description
BS_SP_GEN_ADD_PORG	A form UIBB for displaying the segment “additional purchasing organization” data of the vendor’s general data.

ABAP Object List: Company Codes

The genIL object defining the field catalogue is `SP_CompanyCode`.

Class `CL_BS_SP_COMPANY_CODES` is the responsible feeder class for the list UIBB.

`BS_SP_COMPANY_CODES`. In the standard UI the list is part of the vendor’s general data edit page. It shows the company code key and description only. It is read-only since the actual data is maintained by forms on a specific edit page.

Class `CL_BS_SP_COMPANY_CODE` is the responsible feeder class for the below mentioned form UIBBs.

Form UIBB	Description
BS_SP_COMPANY_CODE	A form for displaying the key fields of vendor’s company code.
BS_SP_COMPANY_CODE_ALL	A form consisting of all segments of the vendor’s company code. This form is currently not used by the SAP delivered UI configurations. It could be used to replace the multiple single form UIBBs, e.g. in case of performance issues.
BS_SP_COMPANY_CODE_ACCOUNTING	A form for displaying the “accounting information” segment of the vendor’s company code.
BS_SP_COMPANY_CODE_AUTO_PAY	A form for displaying the “automatic payment transactions” segment of the vendor’s company code.
BS_SP_COMPANY_CODE_BLOCK_DEL	A form for the "blocking and deletion flag" segments of the vendor’s company code.
BS_SP_COMPANY_CODE_CORRESP	A form for the "correspondence" segment of the vendor's company code data.
BS_SP_COMPANY_CODE_DUNNING	A form for the "dunning data" segment of the vendor's company code data.
BS_SP_COMPANY_CODE_INTEREST	A form for displaying the “interest calculation data” segment of the vendor’s company code.
BS_SP_COMPANY_CODE_INVOICE	A form for the "invoice verification" segment of the vendor's company code data.
BS_SP_COMPANY_CODE_PAYMENT	A form for the "payment transactions" segment of the vendor's company code data.

Form UIBB	Description
BS_SP_COMPANY_CODE_REFERENCE	A form for the "reference data" segment of the vendor's company code data.
BS_SP_COMPANY_CODE_WITHHOLDING	A form for the "withholding tax" segment of the vendor's company code data.

ABAP Object List: Dunning Areas

The genIL object defining the field catalogue is **SP_DunningArea**.

Class **CL_BS_SP_DUNNING_AREAS** is the responsible feeder class for the list UIBB **BS_SP_DUNNING_AREAS**. In the standard UI the list is part of the vendor's company code edit page. It shows the dunning area key and description only. It is read-only since the actual data is maintained by a form on a specific edit page.

Class **CL_BS_SP_DUNNING_AREA** is the responsible feeder class for the form UIBB **BS_SP_DUNNING_AREA**.

ABAP Object List: Extended Withholding Tax Types

The genIL object defining the field catalogue is **SP_WithholdingTaxType**.

Class **CL_BS_SP_WHTAXES** is the responsible feeder class for the list UIBB **BS_SP_WITHHOLDING_TAXES**. In the standard UI the list is part of the vendor's company code edit page. It shows the extended withholding tax type key and description only. It is read-only since the actual data is maintained by a form on a specific edit page. Maintenance in general is only allowed if the parent company code uses the extended withholding tax functionality.

Class **CL_BS_SP_WHTAX** is the responsible feeder class for the form UIBB **BS_SP_WITHHOLDING_TAX**.

ABAP Object List: Purchasing Organization

The genIL object defining the field catalogue is **SP_PurchasingOrg**.

Class **CL_BS_SP_PURCH_ORGS** is the responsible feeder class for the list UIBB **BS_CU_PURCH_ORGS**. In the standard UI the list is part of the vendor's general data edit page. It shows the purchasing organization key and description only. It is read-only since the actual data is maintained by forms on a specific edit page.

Class **CL_BS_SP_PURCH_ORG** is the responsible feeder class for the below mentioned form UIBBs.

Form UIBB	Description
BS_SP_PURCH_ORG	A form for displaying the key field of the vendor's purchasing organization.

Form UIBB	Description
BS_SP_PURCH_ORG_ALL	A form consisting of all segments of the vendor's purchasing organization. This form is currently not used by the SAP delivered UI configurations. It could be used to replace the multiple single form UIBBs, e.g. in case of performance issues.
BS_SP_PURCH_ORG_BLOCK_DEL	A form for the "blocking and deletion flag" segment of the vendor's purchasing organization.
BS_SP_PURCH_ORG_CONDITIONS	A form for the "conditions" segment of the vendor's purchasing organization.
BS_SP_PURCH_ORG_CONTROL	A form for the "control data" segment of the vendor's purchasing organization.
BS_SP_PURCH_ORG_MATERIAL	A form for the "default data material" segment of the vendor's purchasing organization.
BS_SP_PURCH_ORG_SALES	A form for the "sales data" segment of the vendor's purchasing organization.
BS_SP_PURCH_ORG_SERVICE	A form for the "service data" segment of the vendor's purchasing organization.

ABAP Object List: Different Purchasing Data

The genIL object defining the field catalogue is **SP_PurchasingOrg2**.

Class **CL_BS_SP_PURCH_ORGS2** is the responsible feeder class for the list UIBB **BS_CU_PURCH_ORGS2**. In the standard UI, the list is part of the vendor's purchasing organization edit page. It shows the different purchasing data keys and description only. It is read-only since the actual data is maintained by forms on a specific edit page.

Class **CL_BS_SP_PURCH_ORG2** is the responsible feeder class for the below mentioned form UIBBs.

Form UIBB	Description
BS_SP_PURCH2_ORG	A form for displaying the key fields of the vendor's different purchasing data.
BS_SP_PURCH2_ORG_ALL	A form consisting of all segments of the vendor's different purchasing data. This form is currently not used by the SAP delivered UI configurations. It could be used to replace the multiple single form UIBBs, e.g. in case of performance issues.

Form UIBB	Description
BS_SP_PURCH2_ORG_CONDITION	A form for the "conditions" segment of the vendor's different purchasing data.
BS_SP_PURCH2_ORG_CONTROL	A form for the "control data" segment of the vendor's different purchasing data.
BS_SP_PURCH2_ORG_MATERIAL	A form for the "default data material" segment of the vendor's different purchasing data.
BS_SP_PURCH2_ORG_SALES	A form for the "sales data" segment of the vendor's different purchasing data.
BS_SP_PURCH2_ORG_SERVICE	A form for the "service data" segment of the vendor's different purchasing data.

ABAP Object List: Partner Functions

The genIL object defining the field catalogue is **SP_PartnerFunction**.

Class **CL_BS_SP_FUNCTIONS** is the responsible feeder class for the list UIBB **BS_SP_FUNCTIONS**. In the standard UI the list is part of the vendor's purchasing organization edit page. It shows the complete partner function data. The data is maintained within the list. There's no specific edit page.

Class **CL_BS_SP_FUNCTIONS_DIA** is the responsible feeder class for the form UIBB **BS_SP_FUNCTIONS_DIA**. This form is used on a pop-up for the maintenance of partner functions for different purchasing data. It consists of the related additional key fields.

ABAP Object List: Sub-Ranges

The genIL object defining the field catalogue is **SP_Subrange**.

Class **CL_BS_SP_SUBRANGES** is the responsible feeder class for the list UIBB **BS_SP_SUBRANGES**. In the standard UI the list is part of the vendor's general data edit page. The data is maintained within the list. There's no specific edit page.

ABAP Object List: Vendor-like UI

The Vendor-like UI re-uses most of the Supplier Governance (MDG-S) supplier UI forms and lists. Still, it is necessary to provide additional UIBBs where the common ones cannot be used. The following are UIBBs that replace the common ones:

- Form UIBB **BS_SP_GEN_ADMIN** is used additionally for administrative data of the Vendor Like UI. This form has a specific feeder class **CL_BS_SP_GUIBB_VENDOR**.
- Composite UIBB **BS_SP_DETAILS_CU** is used instead of the common composite UIBB of the supplier.

- Form UIBB **BS_SP_GEN_CONTROL_VL** is used instead of the common form UIBB for the “control data” segment of the supplier.
- Form UIBB **BS_SP_GEN_DELETE_VL** is used instead of the common form UIBB for the “blocking and deletion flags” of the supplier.
- List UIBB **BS_SP_VL_QUERY_RESULT** is used instead of the common list UIBB for supplier search results.

ABAP Object List: Lean Vendor Creation UI

The Lean Vendor Creation UI re-uses most of the Supplier Governance (MDG-S) supplier UI forms and lists (the major part of the UIBBs are hidden). Still, it is necessary to provide additional UIBBs where the common ones cannot be used. The following are UIBBs that replace the common ones:

- Composite UIBB **BS_SP_LVC_DETAILS_CU** is used instead of the common composite UIBB of the supplier.
- Form UIBB **BS_SP_LVC_ORG_UNITS** is an additional form UIBB to allow the request for a company code and purchasing organization. It consists of the key fields only. This form has a specific feeder class **CL_BS_SP_GUIBB_LVC_ORG_UNITS**.
- List UIBB **BS_SP_LVC_QUERY_RESULT** is used instead of the common list UIBB for supplier search results.

Additional Information

Links

[Customer Vendor Integration on SAP Help](#)

[FPM Cookbook on SDN](#)

[MDG Guides for EhP 6 on Service Market Place](#)

[MDG Extensibility Options on SDN](#)

[Extensibility Options for SAP Master Data Governance](#) -> [Customer / Supplier Data](#) -> [Data Model Metadata](#)

How-to Guides

[Extensibility Options for SAP Master Data Governance](#) -> [Customer / Supplier Data](#)

- [How-to Guide: Create and Register a Custom Handler Class](#)
- [How-to Guide: Create or Redefine a UI Feeder Class](#)
- [How-to Guide: Extend MDG-S / MDG-C Data Model by a New Entity Type \(Flex Option\)](#)
- [How-to Guide: Extend MDG-S / MDG-C Data Model by a New Entity Type \(Reuse Option\)](#)
- [How-to Guide: Extend MDG-S / MDG-C Data Model by a New Field \(Reuse Option\)](#)
- [How-to Guide: Default the ERP Vendor Creation with MDG-S](#)
- [How-to Guide: Filter an Input Help for a Business System](#)

SAP Notes

- 1637249 specifying required information for OSS support

Copyright

© Copyright 2013 SAP AG. 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 z10, 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.