SAP S/4 HANA MASTER DATA GOVERNANCE

2021-03

MASTER DATA GOVERNANCE, Custom Data Model

DEMO EXPERIENCE

Creation of a Custom Data Model for a WBS

Chris Kavanagh

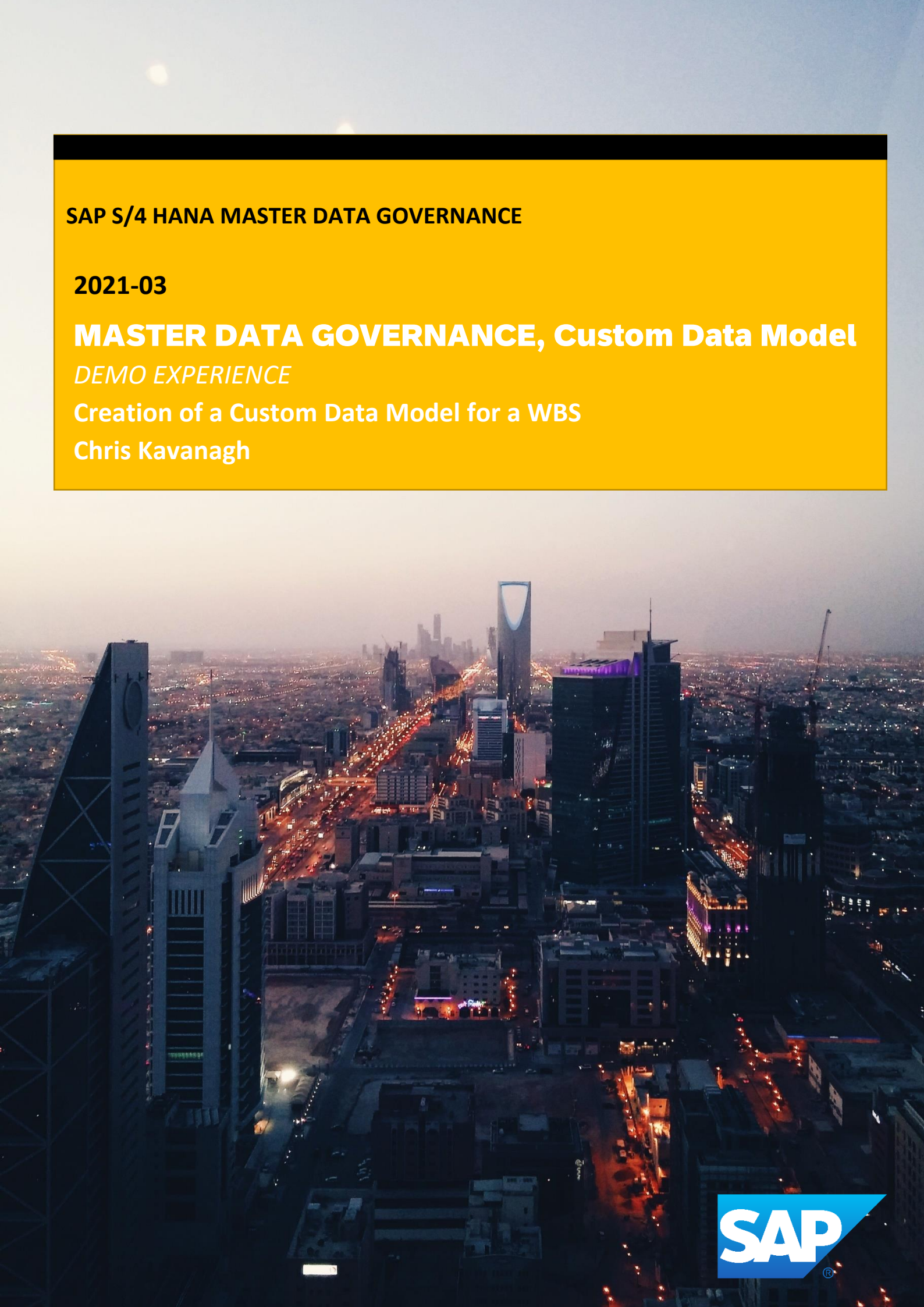


Table of Contents

Creating a Custom Data Model for a Work Breakdown Structure

The SAP MDG custom Object framework is provided by SAP to help users model and build MDG applications for the master data objects specific to their business. It can additionally be used for SAP master data objects that SAP haven't provided any standard MDG applications. For example, there is no out-of-the-box data model and no MDG UI applications delivered for locations, so you could use the Custom Object framework to build an SAP MDG application for locations.

Project System in S/4HANA

Project System (PS) is a project management tool that provides users with support in all phases of their enterprise project. In SAP S/4HANA, PS provides structures that can be used to model and organise projects flexibly

PS provides two structures for mapping an enterprise project:

- Work Breakdown Structures (WBS)
- Networks

A WBS is a model of the project that shows the project deliverables in hierarchical form. WBS's are used to organise a project in the form of a hierarchy and to map the structure of a project. WBS's are made up of WBS elements that are structures in various levels to produce a hierarchy model of the project activities to be carried out.

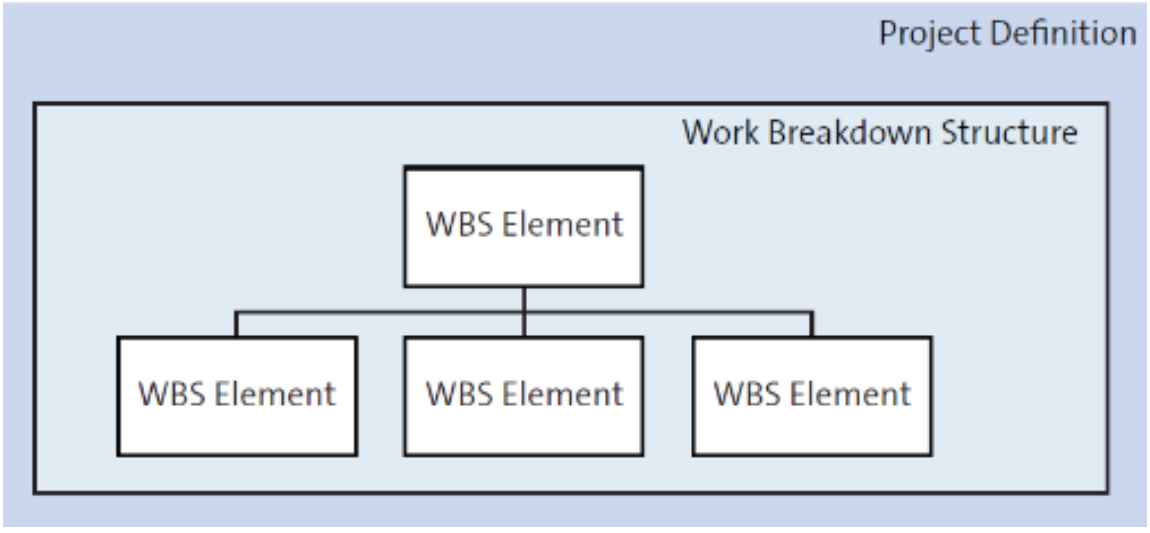
A network represents the course of a project by describing the time sequence and dependencies of events and activities in a project. Networks are used to represent project activities and logical relationships between the project activities.

The following elements are considered master data for the PS application:

- WBS elements
- Networks
- Activities.

Depending on the requirements, you can use WBSs, networks, or both to map your project in the SAP system. However, the scope of this book only includes project definition, WBSs, and WBS elements, not networks and activities

Each WBS can contain multiple WBS elements, as shown in the image below:



The following are different components of WBS

WBS elements

WBS elements represent a work package in an enterprise project. WBS elements are actual elements that are used as account assignment objects to record costs, and they can also be used as planning elements. WBS elements are arranged in a hierarchical manner, allowing the data to be summarized at any level.

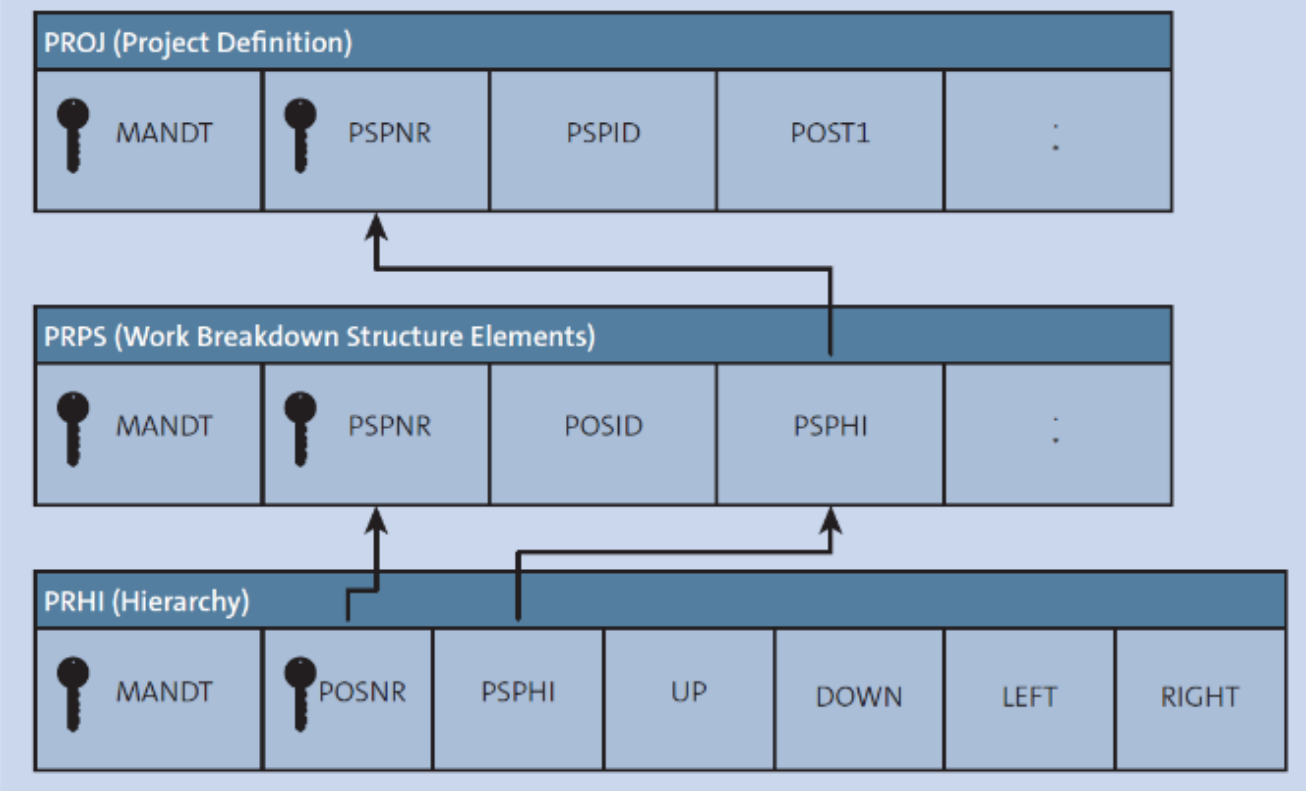
Project definition

The project definition is used to define the common attributes that are shared across the structures (WBS/network) and structure elements (WBS elements/activities) assigned to the project. Project definition is a mandatory component for creating a project with a network, WBSs, or both. The project definition holds the data that affects the whole project. For example, a controlling area entered in the project is applicable for the whole project. Project definition is also used to define organizational data such as company code, business area, profit centre, and plant. This organizational data is defaulted across the WBS elements.

WBS

The WBS is the model of the project and shows the work packages in a hierarchical structure. Each work package in an enterprise project is represented by WBS elements

Below are the various tables that together form the data model for PS. All data is saved in the tables. This database architecture will be the basis for the SAP Master Data Governance data model.



The MDG foundation framework uses the SAP MDG data model entities, attributes of entities, and relationships between entities to generate the staging area. The staging area is an exclusive persistence layer for MDG, generated from an active SAP MDG data model. The staging area is used to store both active data and inactive data. The goal of the data model is to generate these staging area tables correctly and be the single source of information for relationships between various SAP Master Data Governance entities. The MDG data model is also a source of metadata required for UI modelling. There are two storage modes for active data:

Reuse mode

This mode is used if the tables needed already exists in SAP S/4HANA. Usually these are the master data objects that are available as part of the SAP S/4HANA data model but aren't delivered as out-of-the-box SAP MDG data models. To use the reuse active area for the custom data models, you must create an active area and assign the access class to the active area.

Flex mode

This mode is used if no tables are available in SAP S/4HANA. Usually these are the master data objects that aren't available as part of the standard SAP S/4HANA data model. Ideally, the flex option is preferred for business objects that require edition management. SAP S/4HANA doesn't have the edition concept; instead, these layers enforce time dependency by using valid-from date, valid-to date, or both as key fields in the table.

In our example of WBSs in PS, we'll choose flex mode because the flex model doesn't require the creation of an active area access class, and our current data model involves a hierarchy.

Create a Custom Data Model

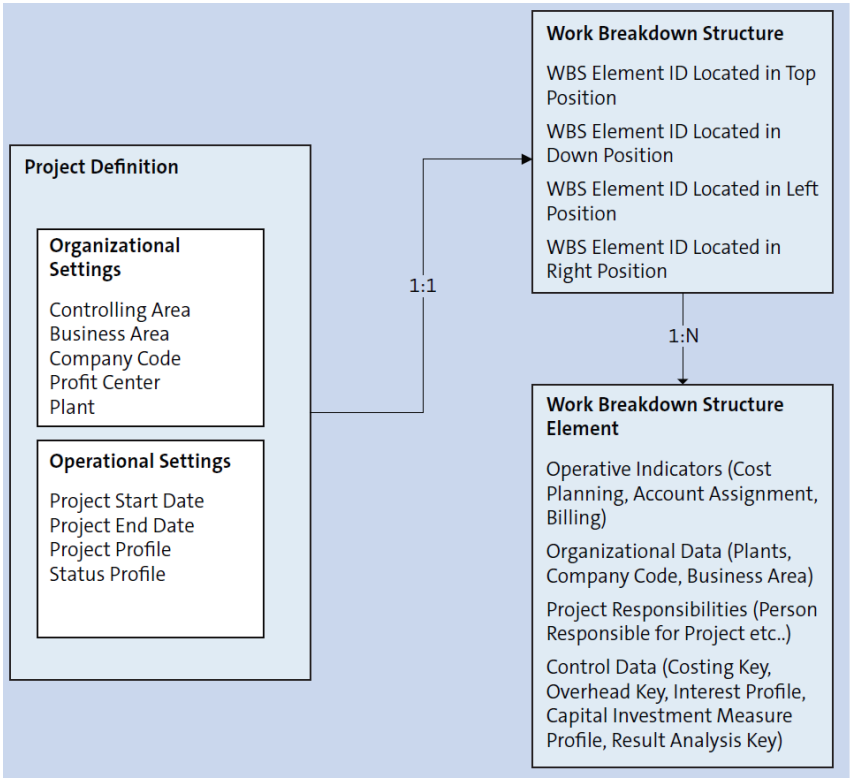
The first step toward building custom SAP MDG applications is to create a custom data model. The Generic Interaction Layer (GenIL) provides uniform API services to access and manipulate underlying business data. The Business Object Layer (BOL) consumes the GenIL API. The following sections explain how to create the custom data model, entities, and relationships in detail.

Concepts and Prerequisites

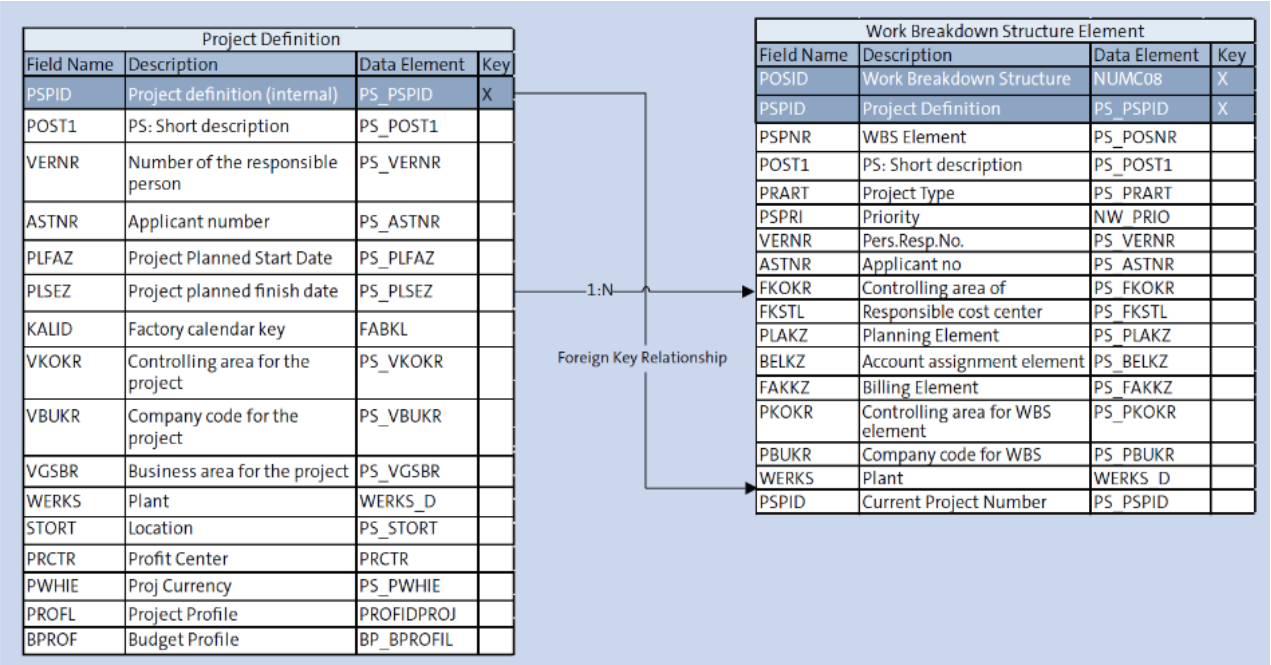
The process of creating entities and attributes for our custom data model ZX will be described via the following:

- Conceptual data model
- Logical data model
- Physical data model

First the conceptual data model: This data model is a high-level representation of the data model architecture. The conceptual model is created with non-technical names make it easy to understand and process for project stakeholders such as executives, business users, and business subject matter experts (SME's). The conceptual data model acts as the basis for creating the logical data model.



The logical data model is a more technical representation of the data model, it displays entities attributes and relationships. The logical data model is normalized by specifying the field-level details such as data type and data length. Below is the logical data model for our custom data model; you can see project definition and WBS elements along with detailed attributes. The target audience for the logical data model, is business SME's, expert modelers, and application experts.



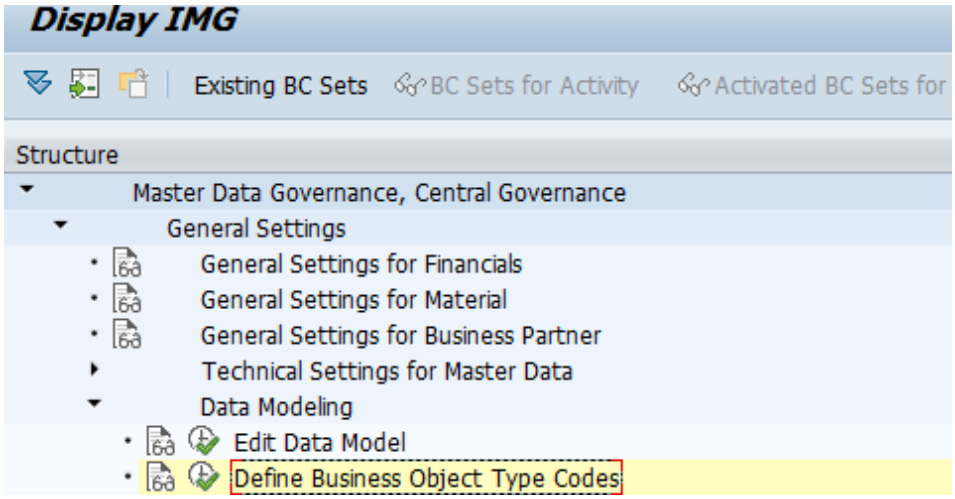
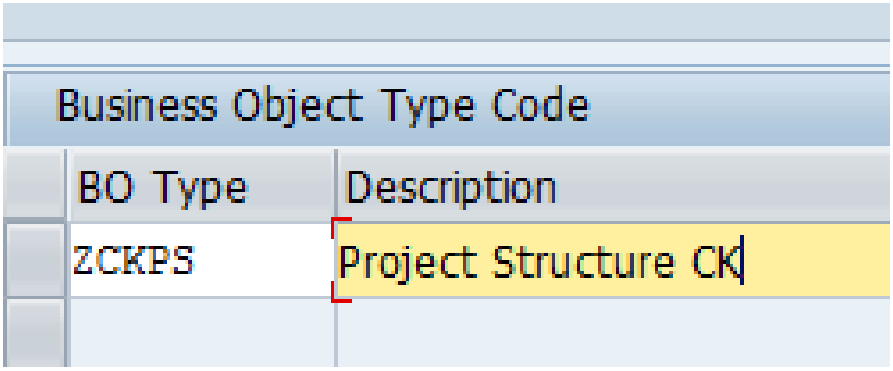
Project Definition			
Field Name	Description	Data Element	Key
PSPID	Project definition (internal)	PS_PSPID	X
POST1	PS: Short description	PS_POST1	
VERNR	Number of the responsible person	PS_VERNR	
ASTNR	Applicant number	PS_ASTNR	
PLFAZ	Project Planned Start Date	PS_PLFAZ	
PLSEZ	Project planned finish date	PS_PLSEZ	
KALID	Factory calendar key	FABKL	
VKOKR	Controlling area for the project	PS_VKOKR	
VBUKR	Company code for the project	PS_VBUKR	
VGSBR	Business area for the project	PS_VGSBR	
WERKS	Plant	WERKS_D	
STORT	Location	PS_STORT	
PRCTR	Profit Center	PRCTR	
PWHIE	Proj Currency	PS_PWHIE	
PROFL	Project Profile	PROFIDPROJ	
BPROF	Budget Profile	BP_BPROFIL	

Work Breakdown Structure Element			
Field Name	Description	Data Element	Key
POSID	Work Breakdown Structure	NUMC08	X
PSPID	Project Definition	PS_PSPID	X
PSPNR	WBS Element	PS_POSNR	
POST1	PS: Short description	PS_POST1	
PRART	Project Type	PS_PRART	
PSPRI	Priority	NW_PRIO	
VERNR	Pers.Resp.No.	PS_VERNR	
ASTNR	Applicant no	PS_ASTNR	
FKOKR	Controlling area of	PS_FKOKR	
FKSTL	Responsible cost center	PS_FKSTL	
PLAKZ	Planning Element	PS_PLAKZ	
BELKZ	Account assignment element	PS_BELKZ	
FAKKZ	Billing Element	PS_FAKKZ	
PKOKR	Controlling area for WBS element	PS_PKOKR	
PBUKR	Company code for WBS	PS_PBUKR	
WERKS	Plant	WERKS_D	
PSPID	Current Project Number	PS_PSPID	

Finally, the physical data modelling is the process of creating the actual data model in the system. During this step, you take the outcomes (e.g., design document) from the preceding two steps and implement them in the system. The physical data model is system specific and deeply technical in nature.

Step by Step Procedure

Create Business Object Type Code (OTC)

Logon with SAP GUI and start transaction MDGIMG. Navigate to general Settings → Data Modeling → Define Business Object Type Codes.	
Select New Entries in the top right corner to create your BO Type. Following the correct naming conventions such as Z*. EG: ZCKPS Fill in description.	
Save Your changes	

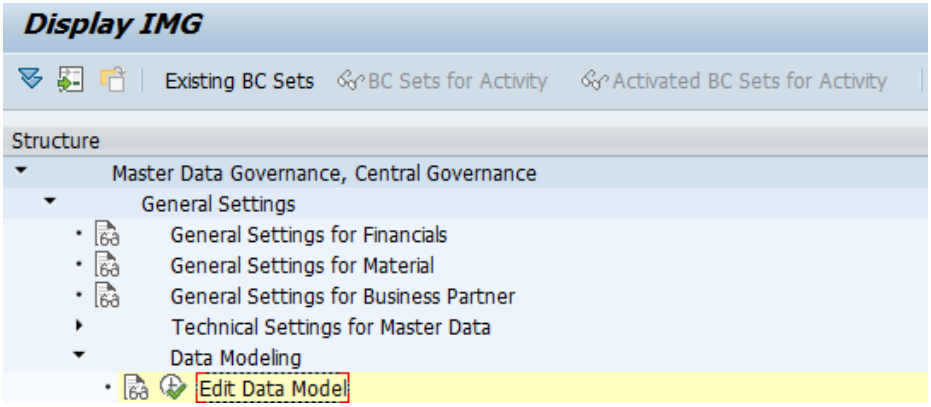
Creating Entities

In the following sections, we'll look at creating entities for your custom data model, starting with type 1 entities before moving on to type 3.

Every data model should have at least one type 1 entity. In the current example, the project definition is one of the type 1 entities.

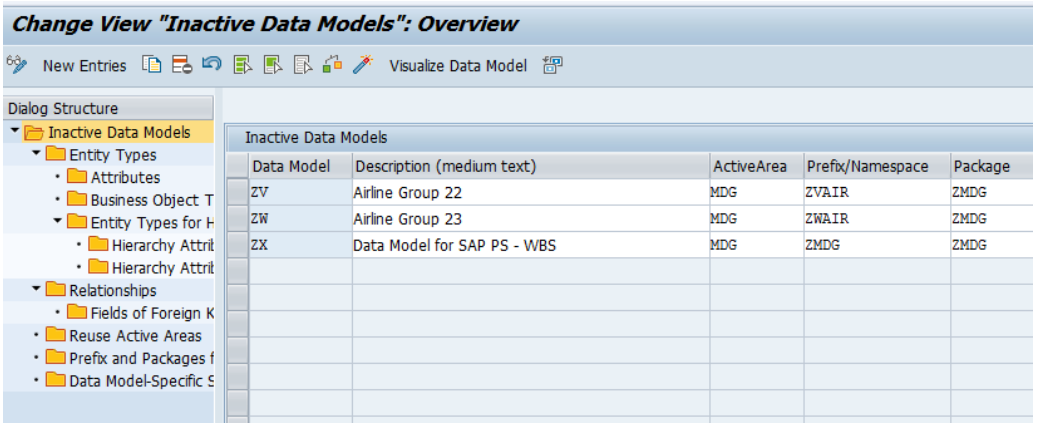
Return to the MDGIMG screen.
Navigate to 'Edit Data Model'.
General Settings →
Data Modeling →
Edit Data Model.

Alternatively, you can use the Configuration workbench



Click New entries to create a new Data Model.

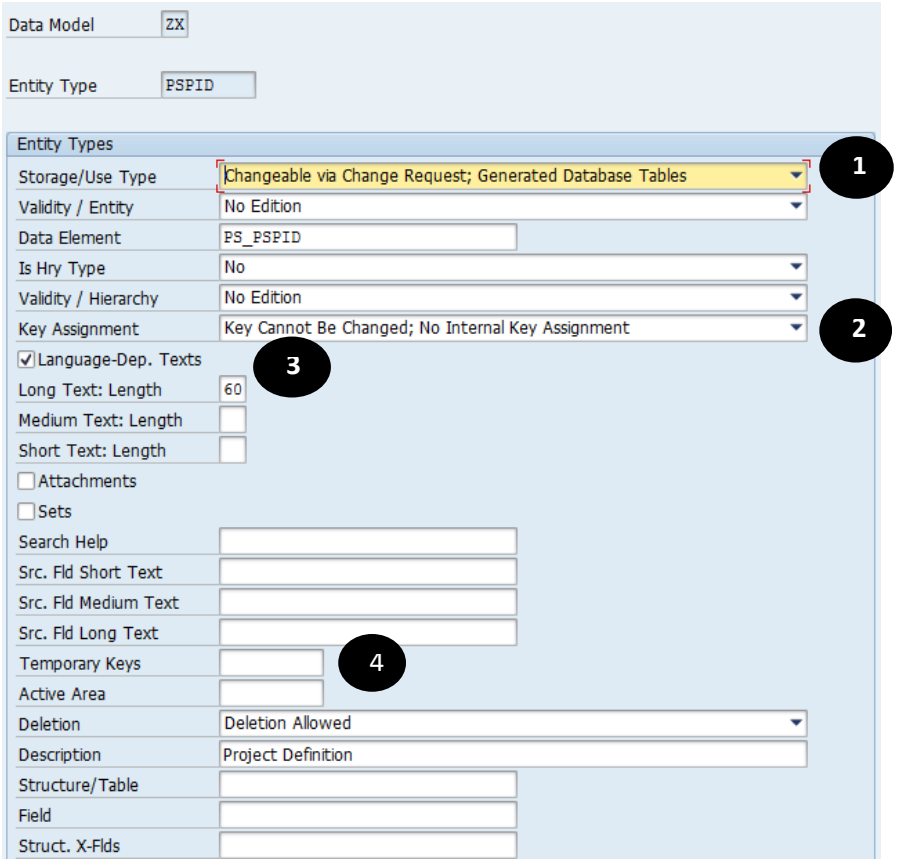
Fill in the details as follows
Data Model: ZX
Descr.: Data Model for SAP PS - WBS
Active Area: provided by MDG
Namespace: ZMDG
Package: ZMDG

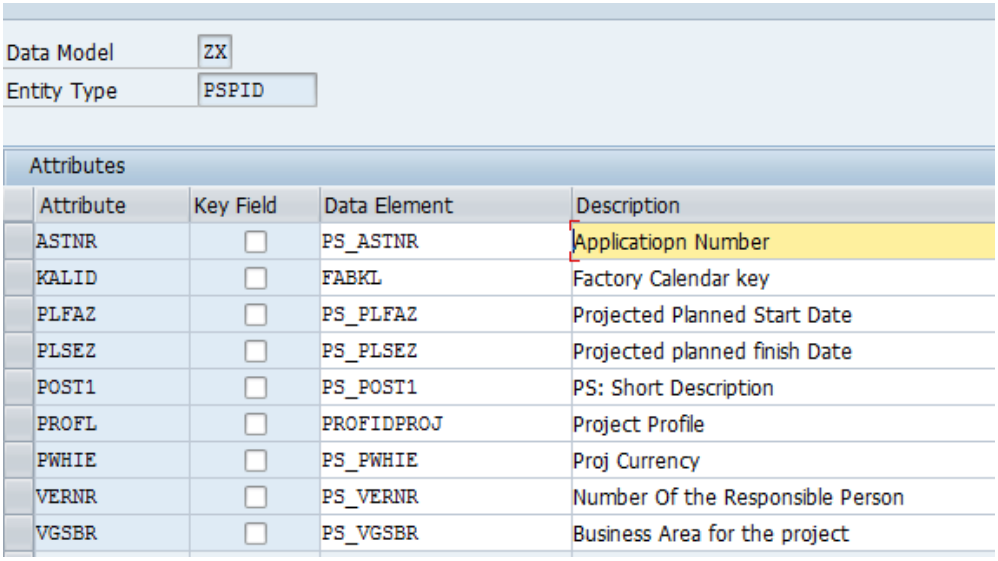
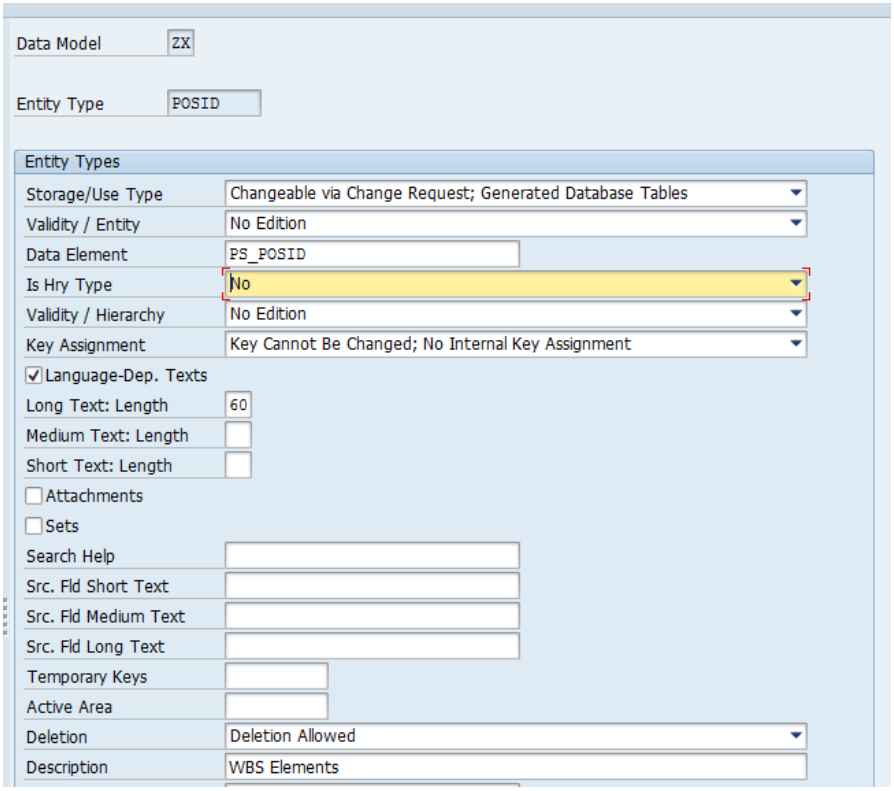


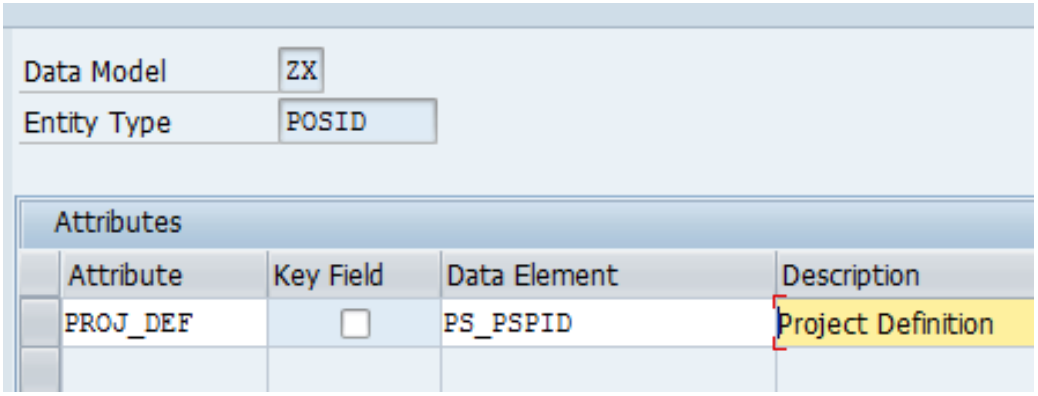
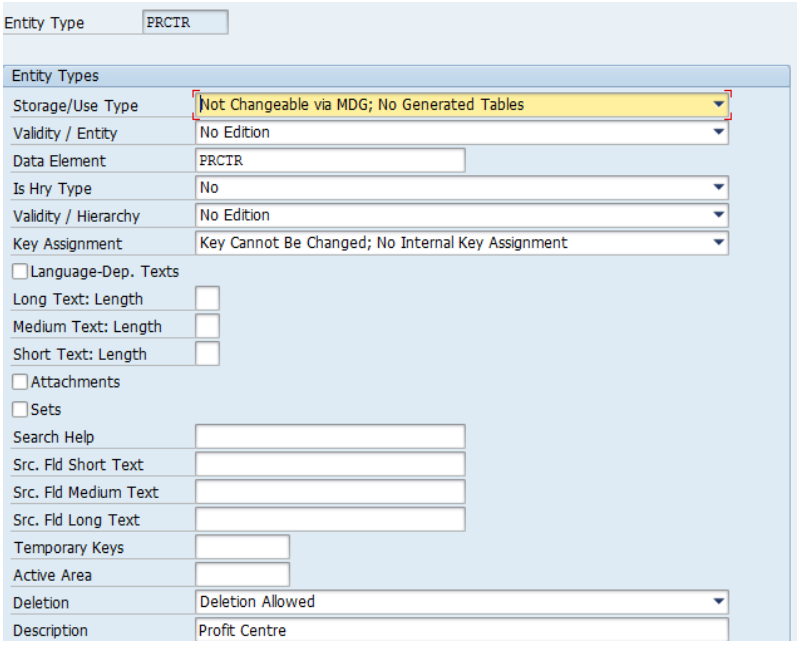
Select new entries and add your first entity PSPID.

Storage: Type 1
Data Element: PS_PSPID

1. Edition management as been disabled. In the underlying SAP S/4HANA data model, PS isn't a time dependent entity
2. Every type 1 entity needs a key assigned.
3. The project definition can have language dependent texts.
4. The Active area field has been left blank; when left blank the active area of the data model is adopted

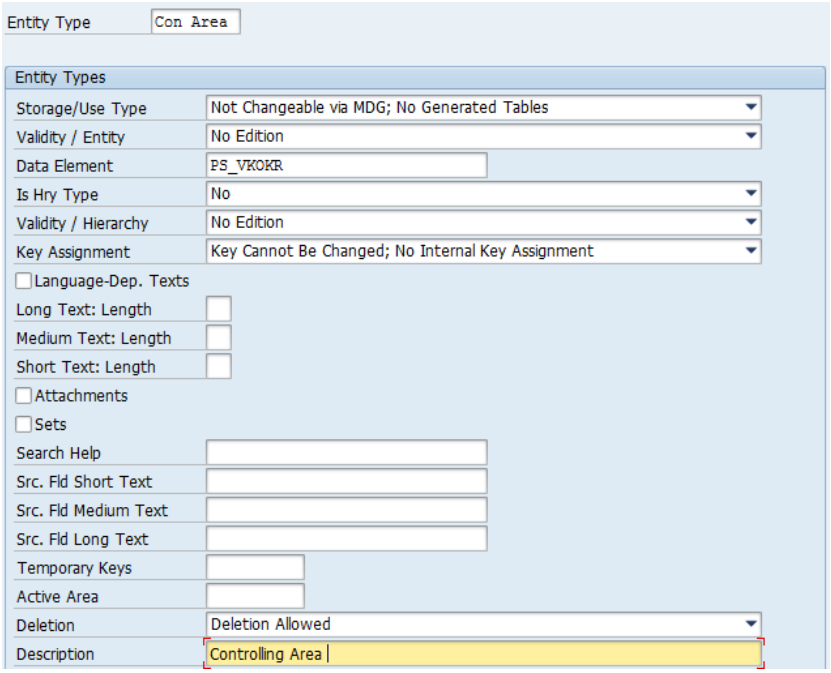


Click on Attributes tab and add New attributes for the domain: See the picture to the right and fill in the attributes as shown.	
Save your changes	
Select new Entries as Previous We'll be creating a WBS element as another type 1 entity, but we will assign it the same business object as the project definition (PSPID). By assigning it to the same business object as the project definition, we're ensuring that a WBS element can't be replicated alone, and it needs to be replicated through the project definition. Fill in the Entity details as shown in the picture	

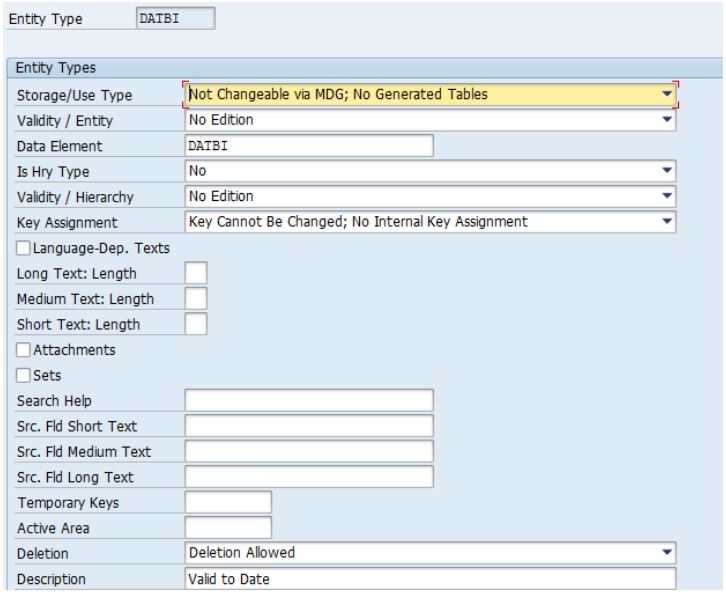
Select attributes We will be adding an attribute PROJ_DEF, this attribute is used to store the project definition ID of the project to which the WBS element is assigned. This attribute needs to have a foreign key relationship with PSPID.	
Save the changes	
The project definition (PSPID) has profit centre as one of the attributes. We cant add profit centre as an attribute directly under the type 1 Entity PSPID because the SAP MDG data model rules dictate that a check table of an entity's attribute cant have more key fields besides the client and key field referring to the attribute.	

The controlling area is present as an attribute both in the project definition and in the WBS element. Therefore, it makes sense to create just one type 3 entity and then assign the same entity as an attribute to the project structure and WBS element entities using referencing relationships.

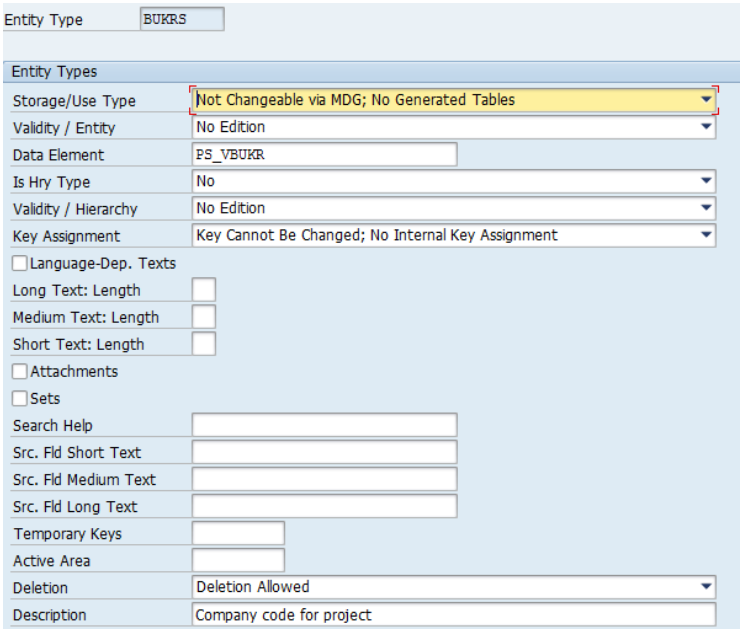
The controlling area is one of the key fields in the check tables assigned to the profit centre therefore, it's imperative that we create the controlling area as a type 2 entity so that the entity can be used to establish a leading relationship.



Like the controlling area the Valid to date is a key attribute in the check table so we will create it as entity 3 so we can establish a leading relationship.

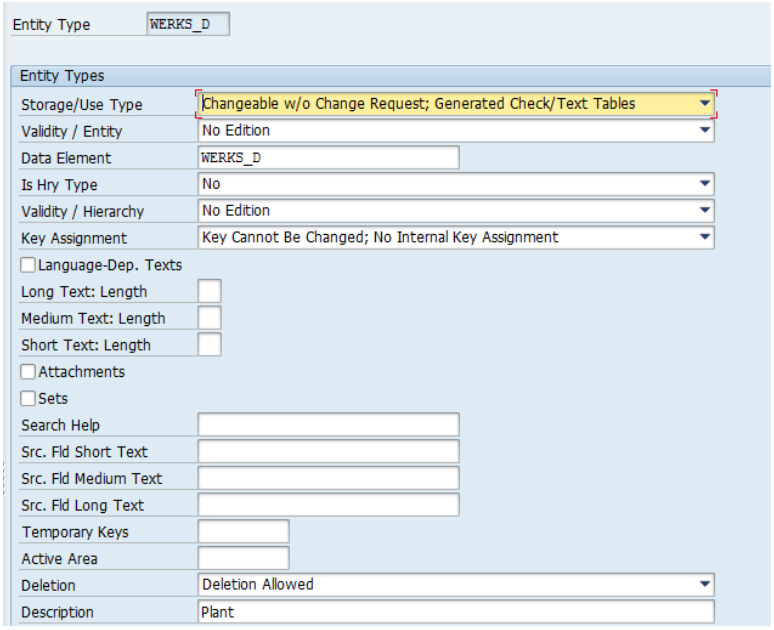


Next, we will create the company code as a type 3 entity as an attribute to PSPID using a referencing relationship. We are choosing Type 3 as we later want to establish it a leading relationship with the plant entity.



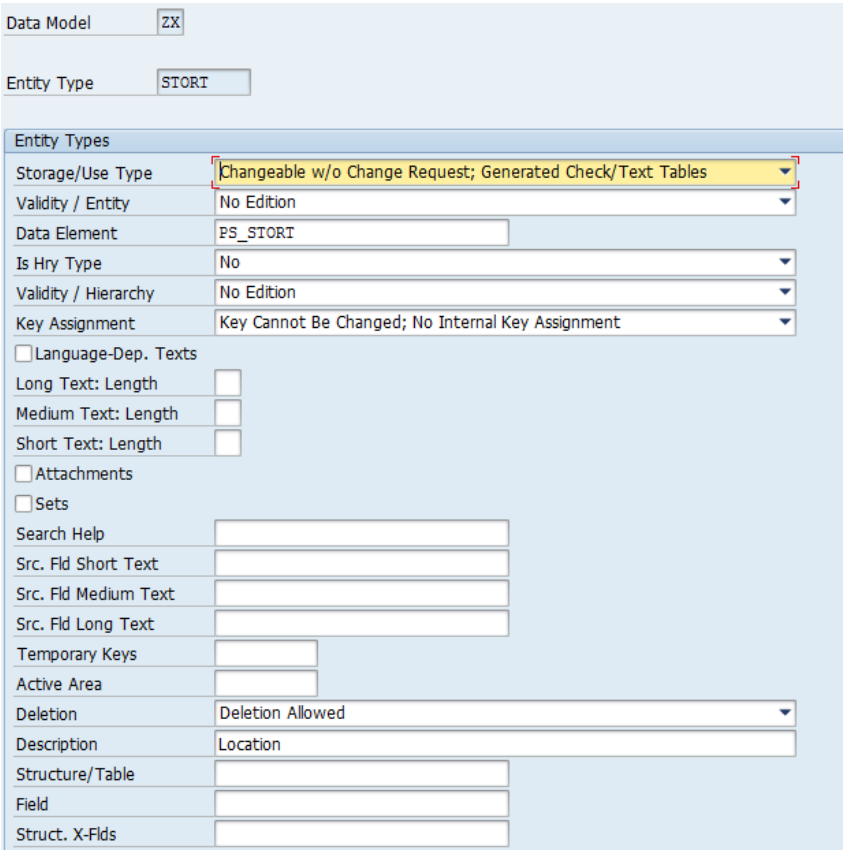
Plant is used to define organization data during project definition.

To ensure that the SAP Master Data Governance framework understands this relationship between plant and location, we'll create plant as a type 3 entity because we need it to be part of the leading relationship for the location attribute.



The check table assigned to location has plant and location as key fields. Therefore, as explained before, to be compliant with the SAP MDG data model rules we must:

-Create a location as type 3 entity
-Create a leading relationship between the entity for plant and the entity for location.
-Create a referencing relationship between the entity for location and entity PSPID

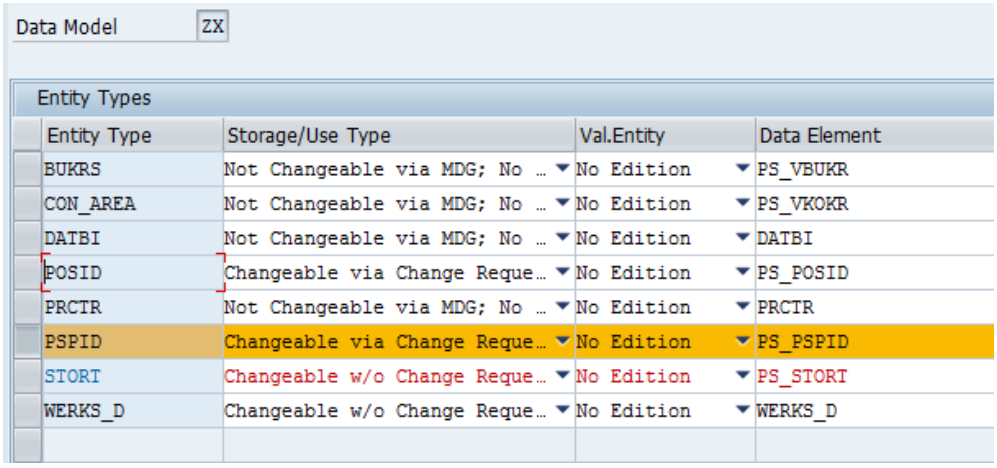


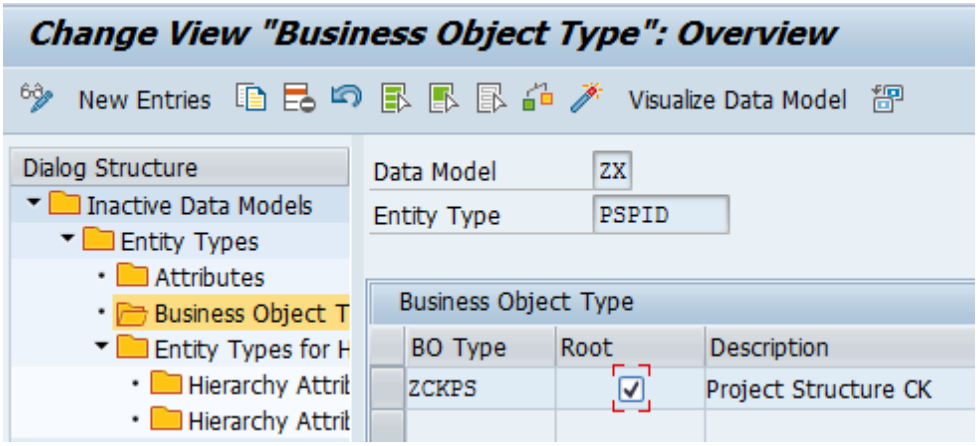
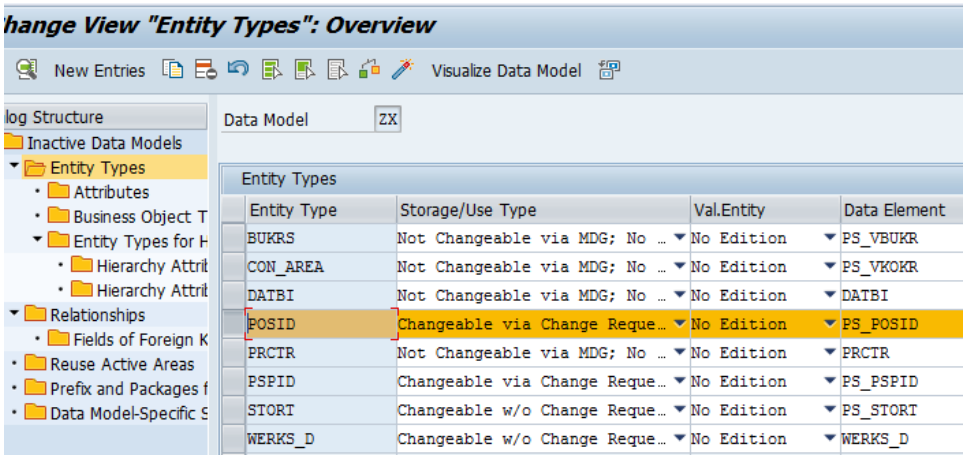
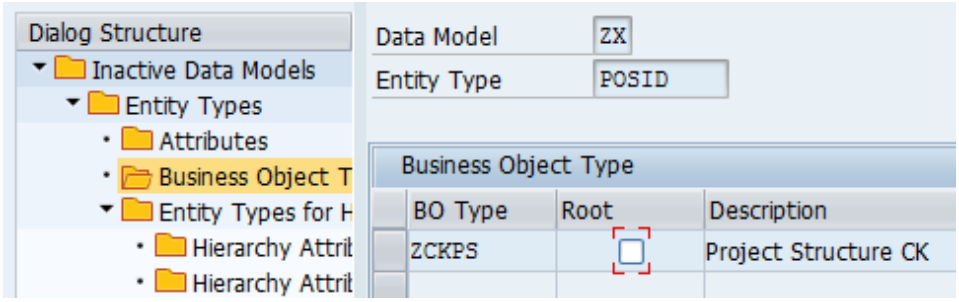
Save your changes and activate

Creating Business Objects

Returning to our list of entities select the first entity we created PSPID.

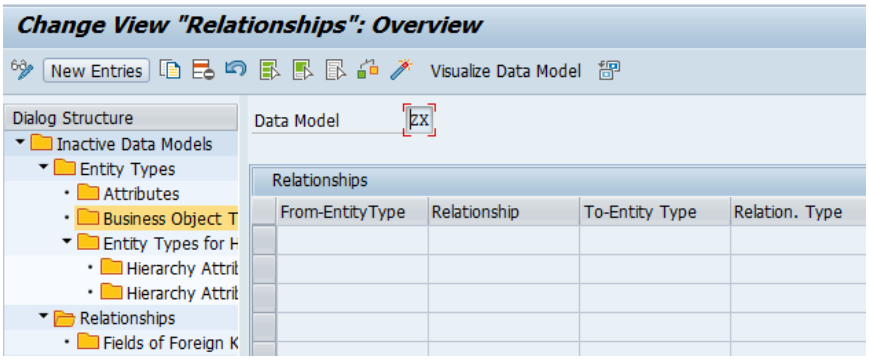
Open Business Object Type view from the left column.



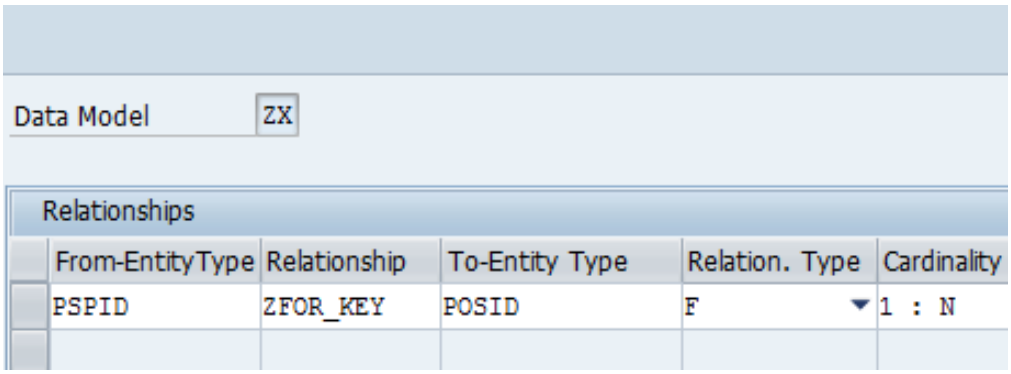
We can input our business object we previously created. Important to note we have selected the root check box for this entity.	
Repeating the steps as previous, return to our list of entities, this select the 2nd entity we created POSID. Then select business object Type from the left column	
Like before input the business object we previously created. In this case leave the root box unchecked as shown	
Save and Activate changes	

Establishing the Relationships

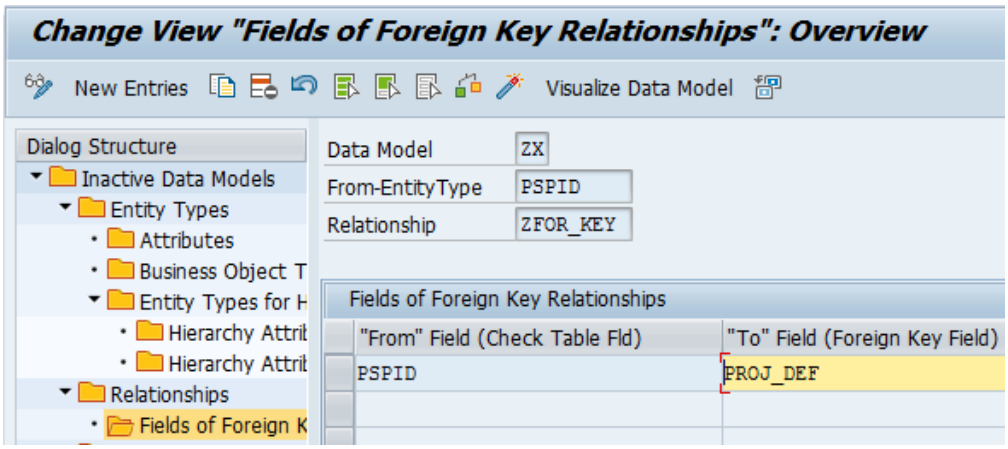
Select Relationships from the left column. Then select new entries from the tool bar to start creating a relationship.



As discussed, before we need a foreign key relationship through attribute PROJ_DEF between PSPID and POSID. Firstly, we will establish the relationship as shown.



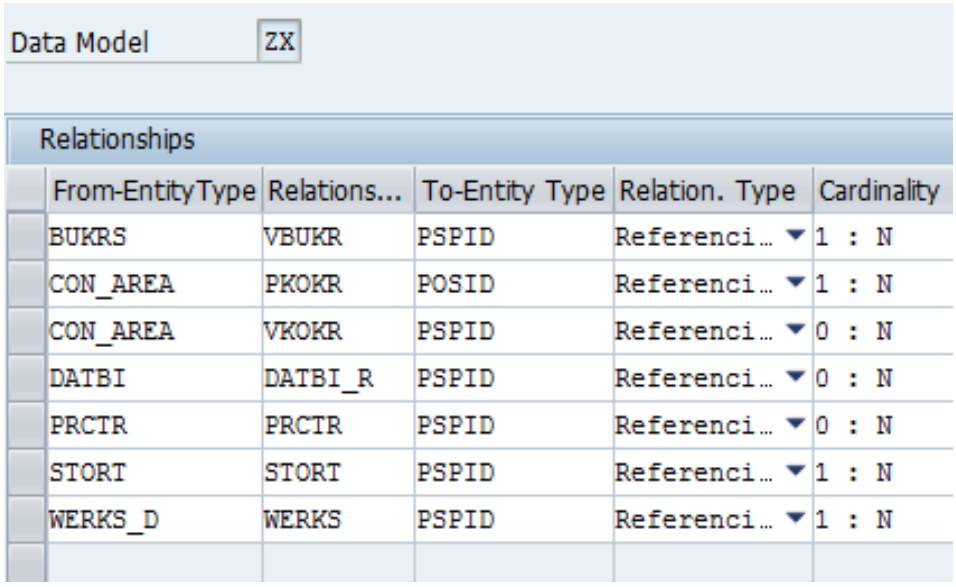
Then select Fields of Foreign keys.
Here is where we will indicate PROJ_DEF is our foreign key. Input as shown



Next, we will establish the referencing relationships. These are the attributes of project definition we previously created.

Like before return to relationships and select new entries from the toolbar.

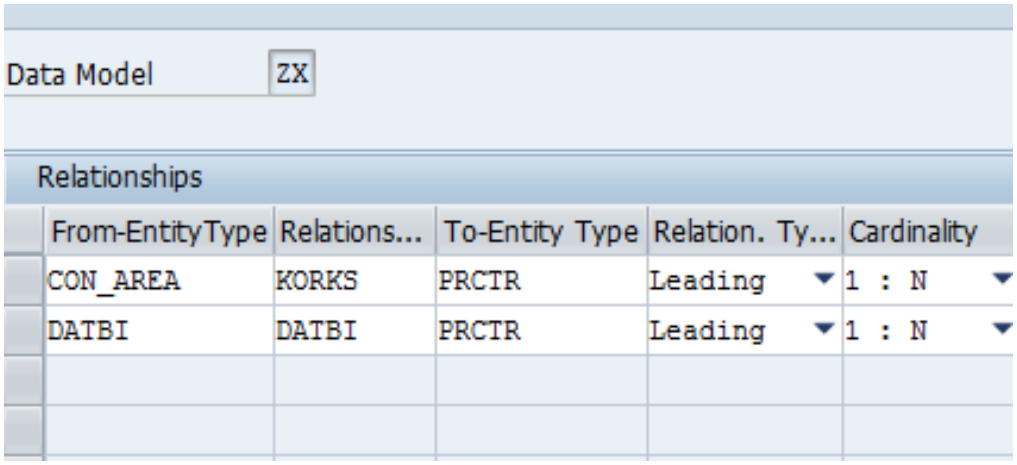
You can enter the relationships as shown.



From-EntityType	Relations...	To-Entity Type	Relation. Type	Cardinality
BUKRS	VBUKR	PSPID	Referenci...	1 : N
CON_AREA	PKOKR	POSID	Referenci...	1 : N
CON_AREA	VKOKR	PSPID	Referenci...	0 : N
DATBI	DATBI_R	PSPID	Referenci...	0 : N
PRCTR	PRCTR	PSPID	Referenci...	0 : N
STORT	STORT	PSPID	Referenci...	1 : N
WERKS_D	WERKS	PSPID	Referenci...	1 : N

Return to the relationships tab and select new entries like before.

Here we are going to establish our leading relationships. Both controlling area and Valid to date are key attributes within the check table of Profit centre. Therefore tis is why we are establishing these relationships.

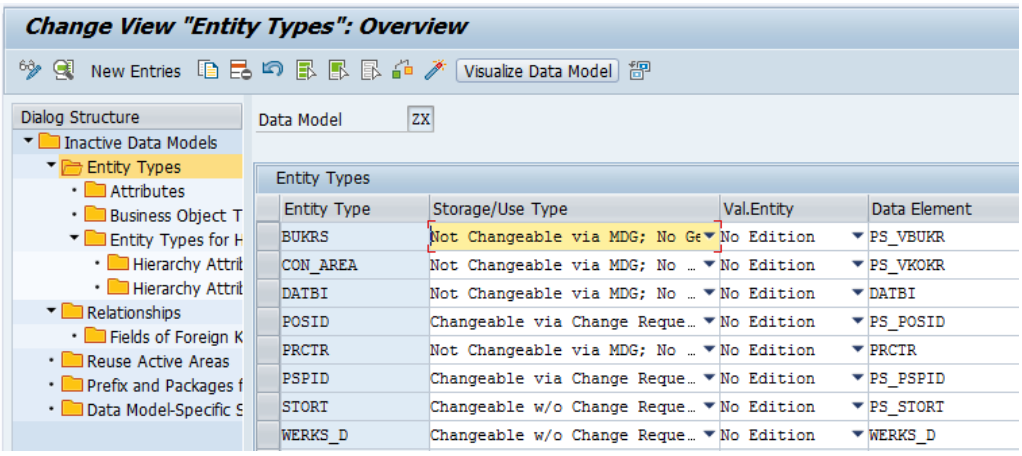


From-EntityType	Relations...	To-Entity Type	Relation. Ty...	Cardinality
CON_AREA	KORKS	PRCTR	Leading	1 : N
DATBI	DATBI	PRCTR	Leading	1 : N

Save and activate

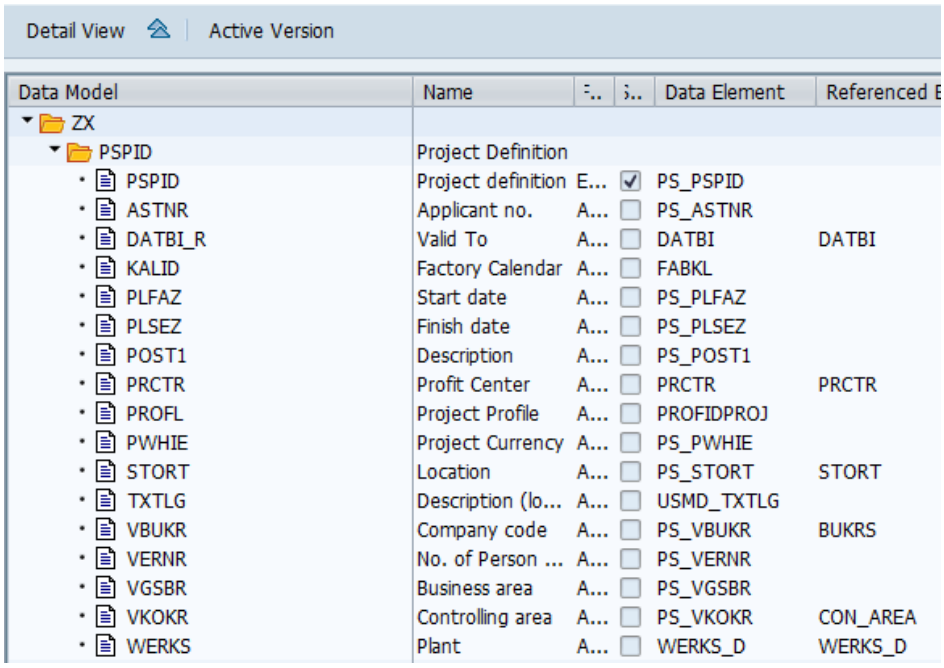
We have now almost completed the data model. We have fully created our PSPID entity.

Select visualize data model from the tool bar to view our Data model structure.



Entity Type	Storage/Use Type	Val.Entity	Data Element
BUKRS	Not Changeable via MDG; No Ge	No Edition	PS_VBUKR
CON_AREA	Not Changeable via MDG; No ...	No Edition	PS_VKOKR
DATBI	Not Changeable via MDG; No ...	No Edition	DATBI
POSID	Changeable via Change Reque...	No Edition	PS_POSID
PRCTR	Not Changeable via MDG; No ...	No Edition	PRCTR
PSPID	Changeable via Change Reque...	No Edition	PS_PSPID
STORT	Changeable w/o Change Reque...	No Edition	PS_STORT
WERKS_D	Changeable w/o Change Reque...	No Edition	WERKS_D

Once opened your data model should appear as shown. You can see the difference relationships that we established.

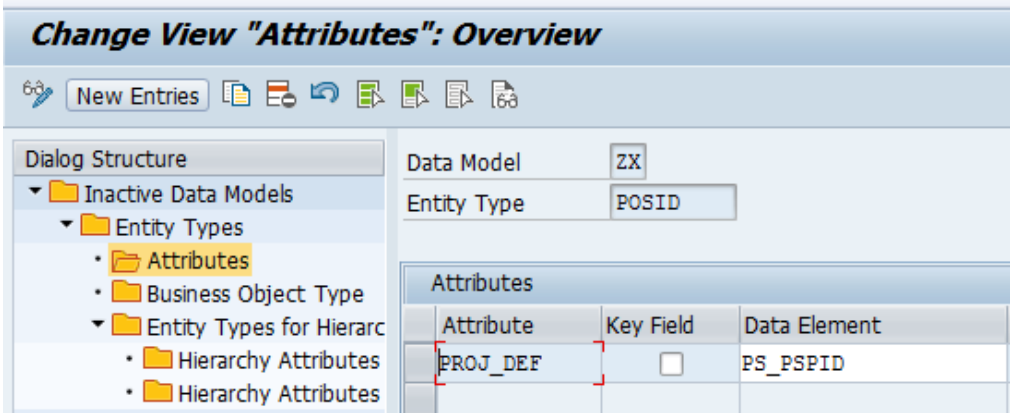


Adding entities to the WBS element entity

We need to add attributes to our POSID entity.

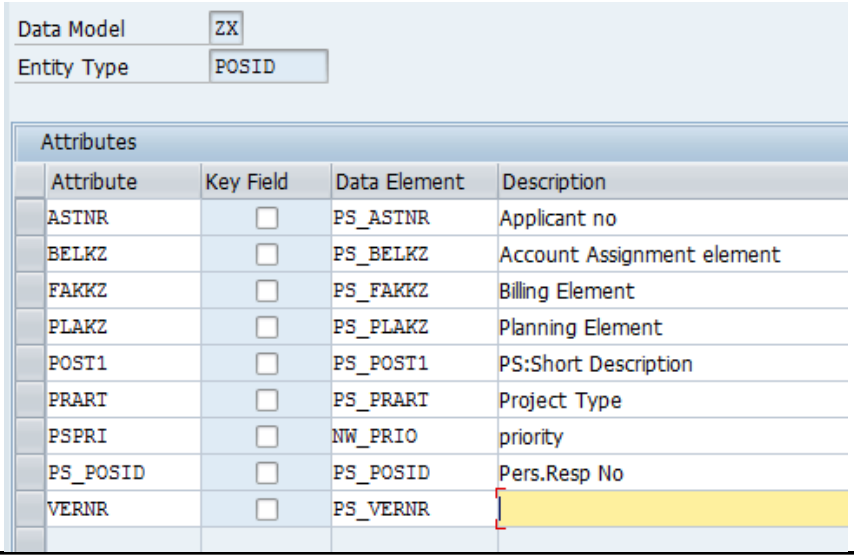
Returning to our entity list select POSID and click attributes from the left column as done previously.

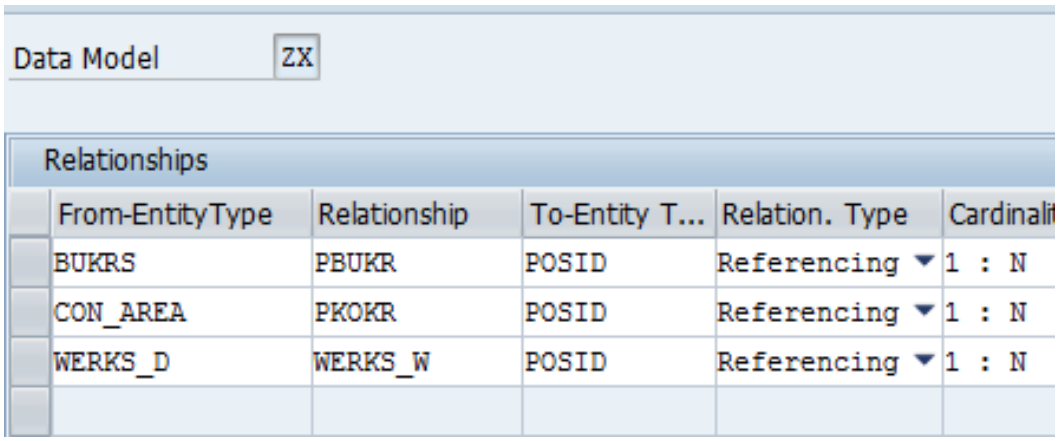
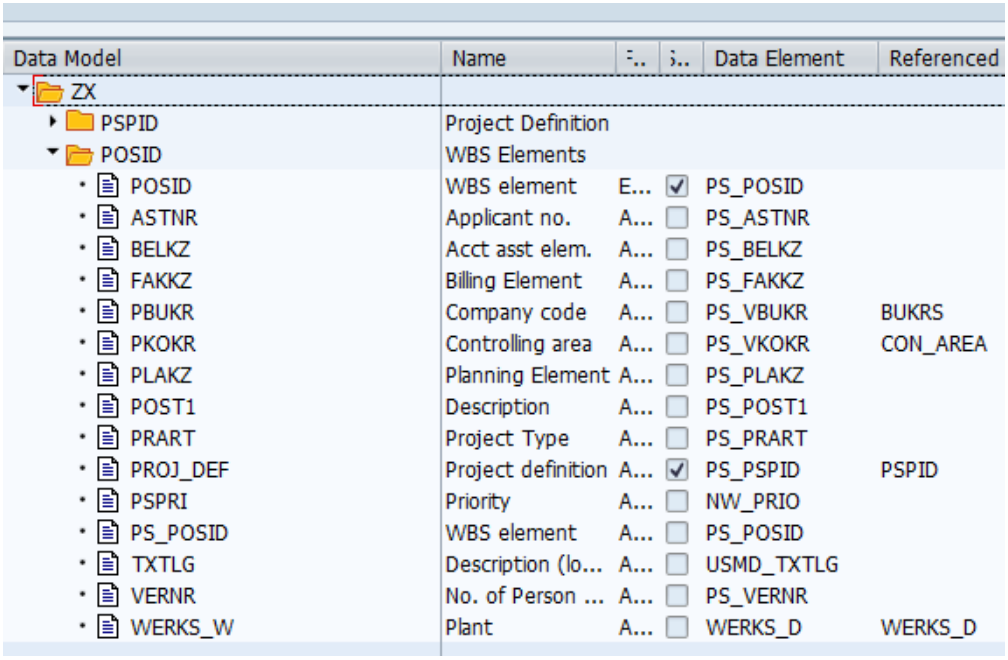
The select new Entries.



Among the attributes that need to be added to the POSID entity there are simple attributes and modelled attributes. Simple attributes can be added through the attribute tab whereas modelled need to be added through relationships.

Shown is the list of simple attributes, add these as shown.



As before to establish relationships, select relationship from the left column and click new entries. Add relationships as shown.	
Save and Activate	
Following the steps as previous select the visualize data model button from the tool bar. Your data model should appear as shown.	

Representing the Work Breakdown Structure Hierarchically

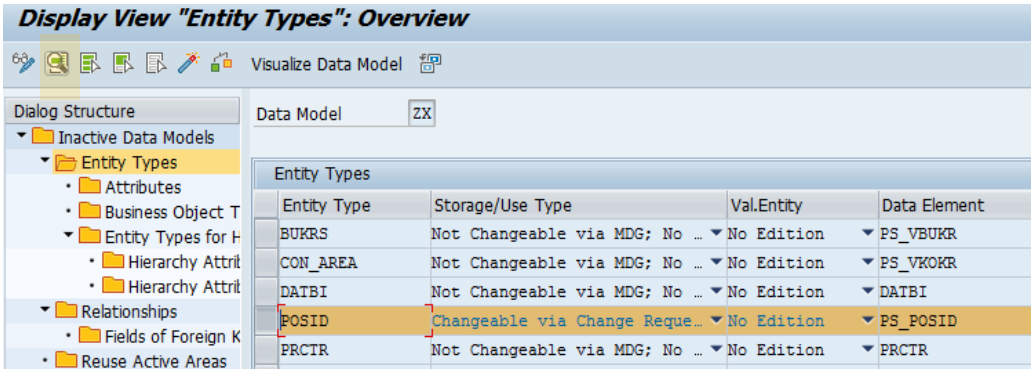
Before we begin, it's important to emphasize that WBSs and WBS elements are different. A WBS is a hierarchical representation of WBS elements, whereas the WBS element represents a work package in a WBS. Entity PSPID represents the project definition, and entity POSID represents the WBS element and not the WBS.

Hierarchy leading entity

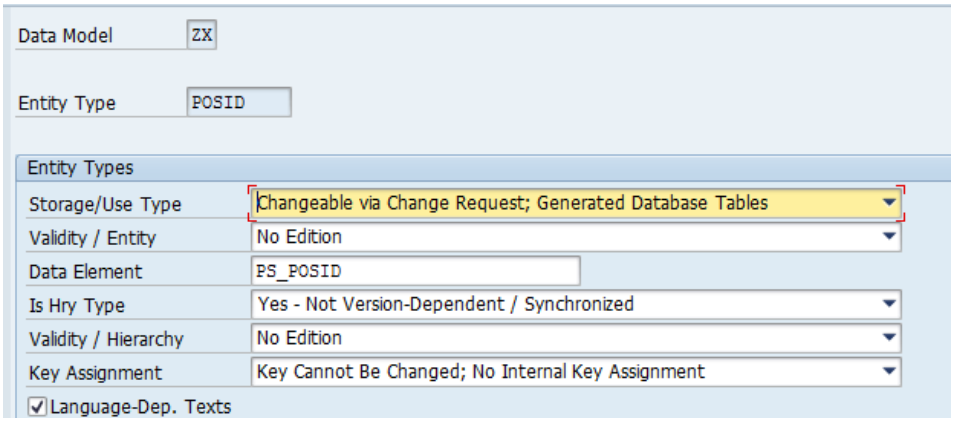
The hierarchy leading entity is the type 1 entity in which the hierarchy setting is activated. In our example, we'll configure entity POSID as the leading entity.

Return to our list of entities and select POSID.

Then select the details button from the toolbar highlight in our picture.

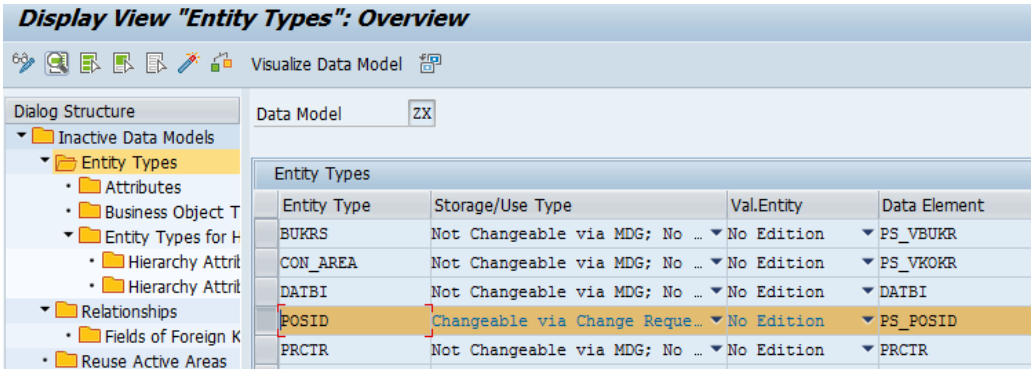


Change the hierarchy type to 'Yes – Not Version-Dependent/ Synchronised'

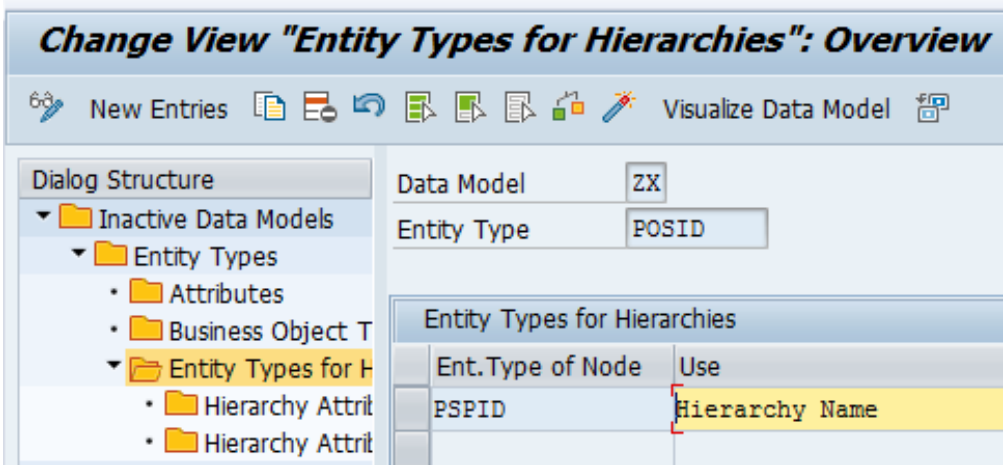


Next we will return to our entity list and select POSID.

This time we will select Hierarchy attributes



Add 'PSPID' and select 'Hierarchy Name' as Entity Use.



Creating a Custom User Interface

This section introduces the concept of building the MDG application UI using the SAP MDG custom object UI framework.

User Interface Framework

The major building blocks of the Custom object UI framework are as follows:

User Interface:
The UI technology for Custom Object UI is Web Dynpro-based Floorplan Manager. Floorplan manager enforces consistency in the UI and compliance with the UI guidelines. By providing generic UI building blocks (GUIBB) and predefined floorplans.

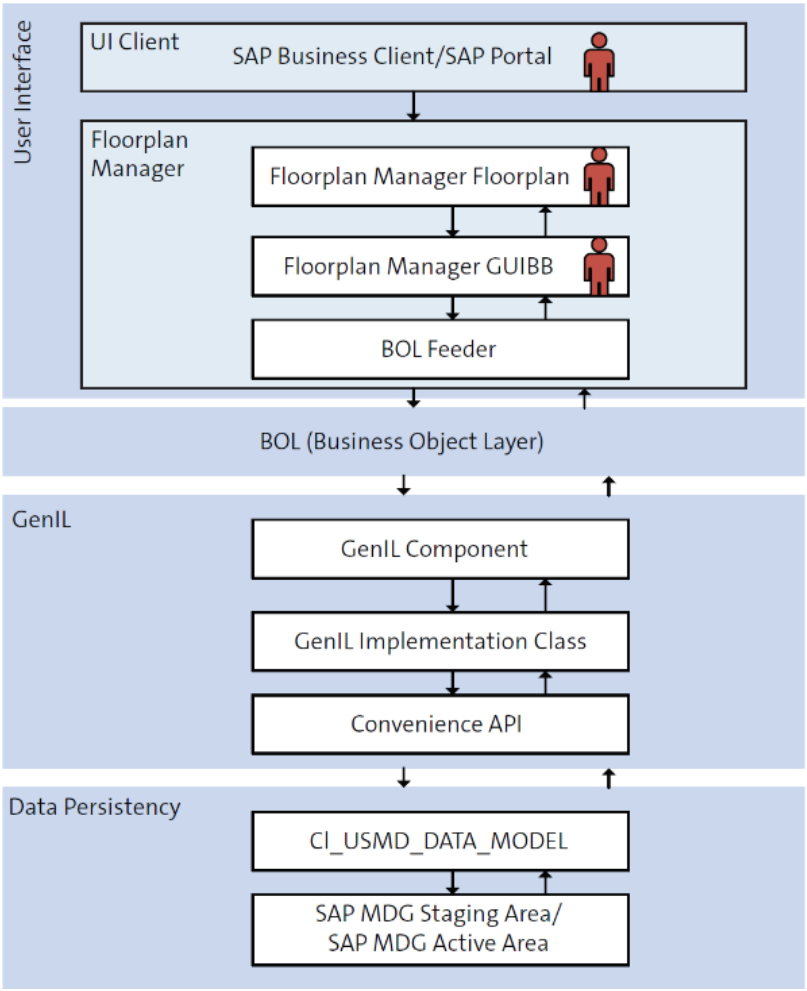
USMD_OVP_GEN is the generic Web Dynpro application delivered by SAP to valid the UI for custom object applications.

GenIL and Business Object Layer (BOL):
The purpose of GenIL us to provide uniform access to the underlying data persistency layer. It encapsulates business object-specific implementation and provides a uniform interface to access data from the persistency layer.

The major building blocks of the custom object UI framework are shown in the image to the right.

This is the runtime architecture and building blocks of the Custom Object UI Framework.

The loose coupling of the various building blocks guarantees minimum disruption. For example, if the SAP OData framework has a BOL adapter, then we can easily re-place the Floorplan Manager with SAP Fiori without changing the code in the underlying GenIL and BOL framework.



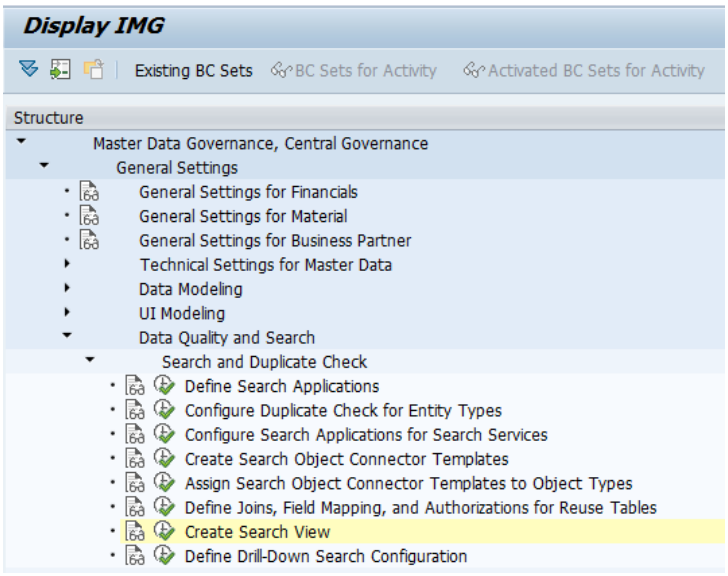
Develop the Search User Interface

SAP Master Data Governance provides a generic search application called USMD_ SEARCH, which can be used to search any data model and type 1 entity using any search provider

Our first step toward enabling the generic search UI is to create a search help.

Run Transaction MDGIMG
General Settings →
Data Quality and Search →
Search and Duplicate Check →
Create Search View.

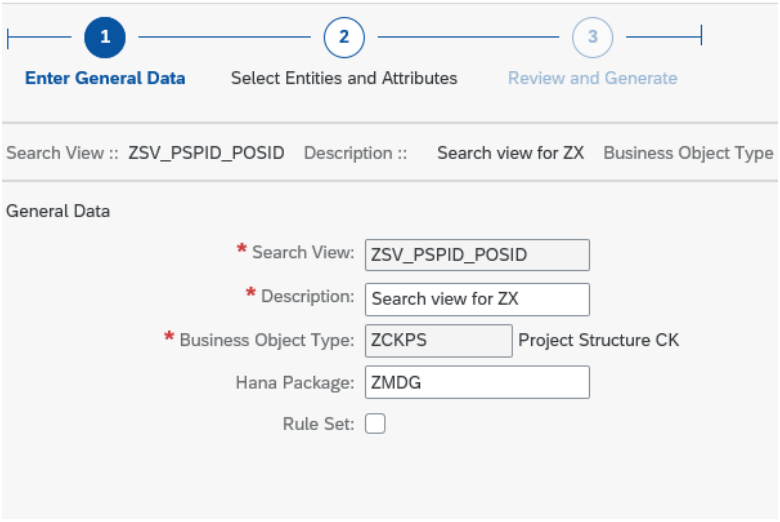
Then select new to create a new search help.



Fill out the general data as shown.

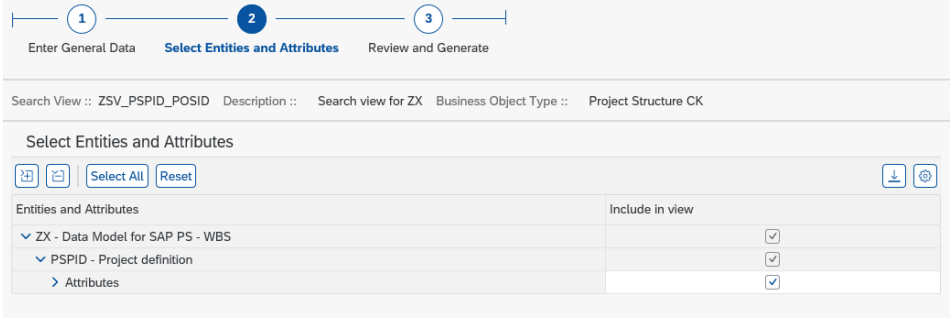
Insert a name in the Search View tab.
A description
Input the business Object we created.

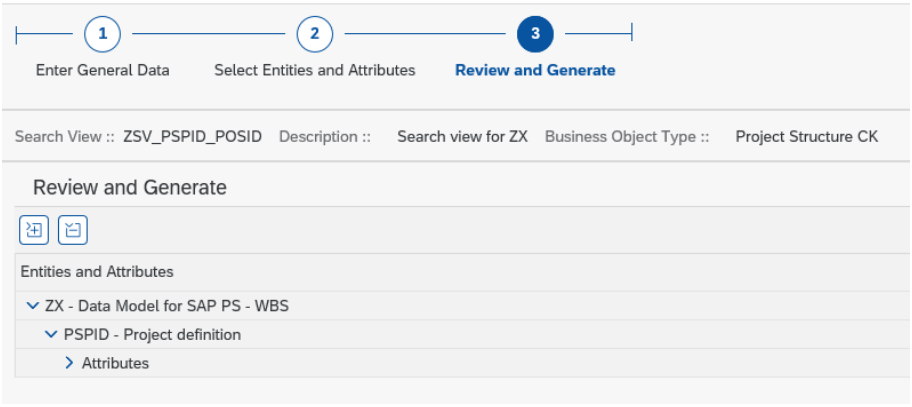
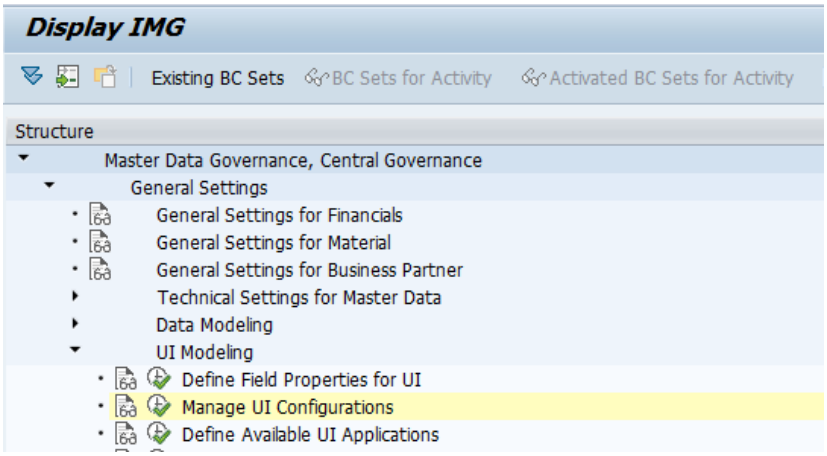
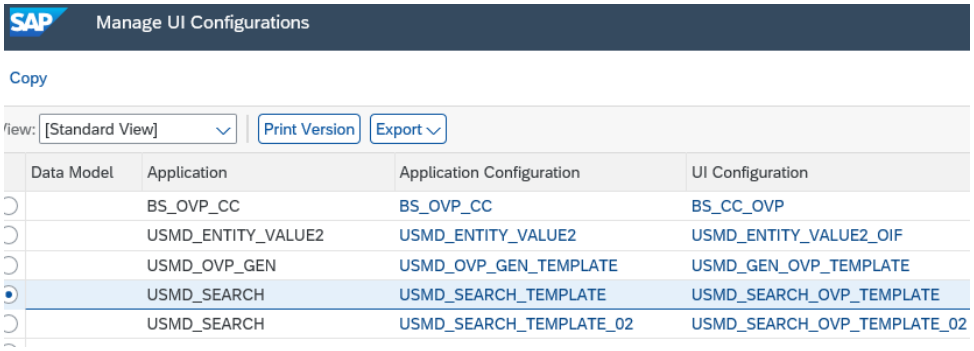
Then select next



Open the PSPID tab and ensure that attributes is checked to be included in view.

Select Next



Check all entities and attributes are included. If so, click save. You will now find your Search view among the list of MDG search views.	
Open the Manage UI Configurations Screen.. General Settings → UI Modeling → Manage UI Configurations. This IMG activity is used to manage all the object maintenance-related UIs	
We are going to create a new generic search application. Select the application configuration USMD_SEARCH_TEMPLATE and click the copy button	

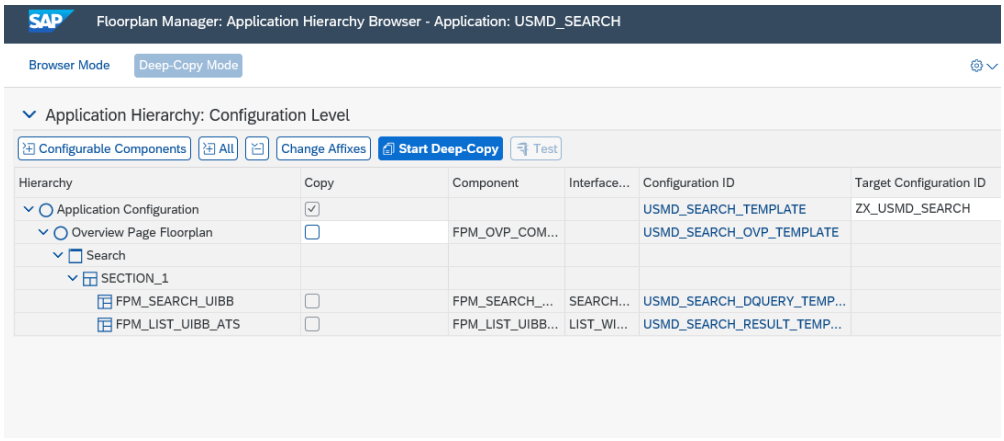
First thing we will do is change the affixes. To do this select Chane Affixes. We will input our data model here ZX.

Select Ok

Next we can rename the Target configuration, by removing the template.

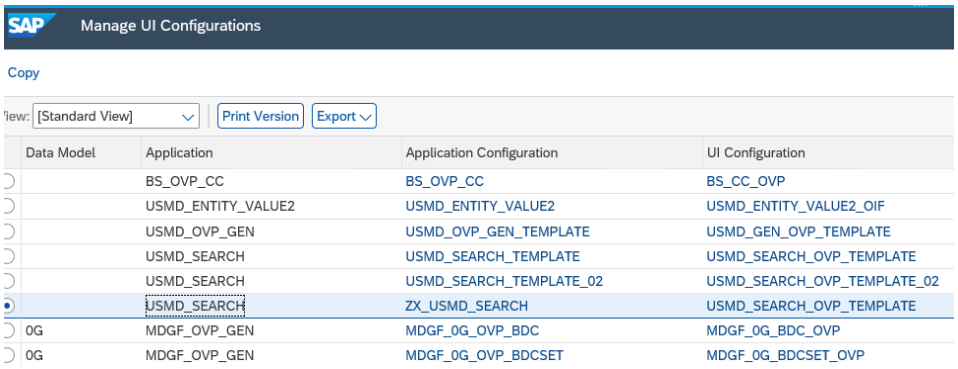
We can also untick the copy box for Overview Page Floorplan.

Select Deep Copy.



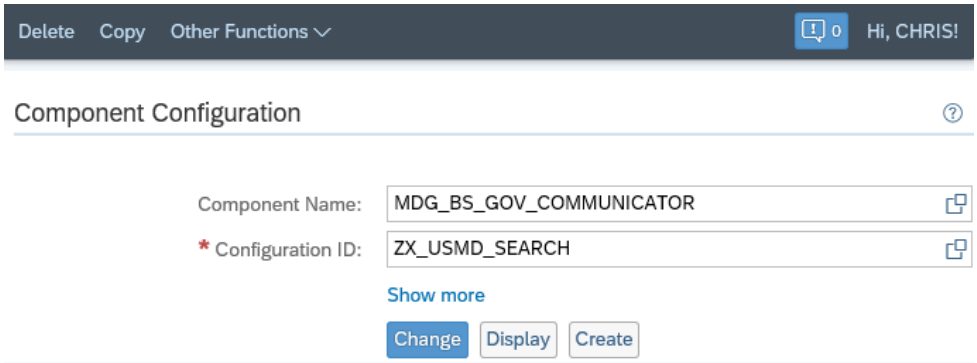
Return to the list of UI configurations, Find the UI configuration we just created.

Then click on the application Configuration link ZX_USMD_SEARCH.



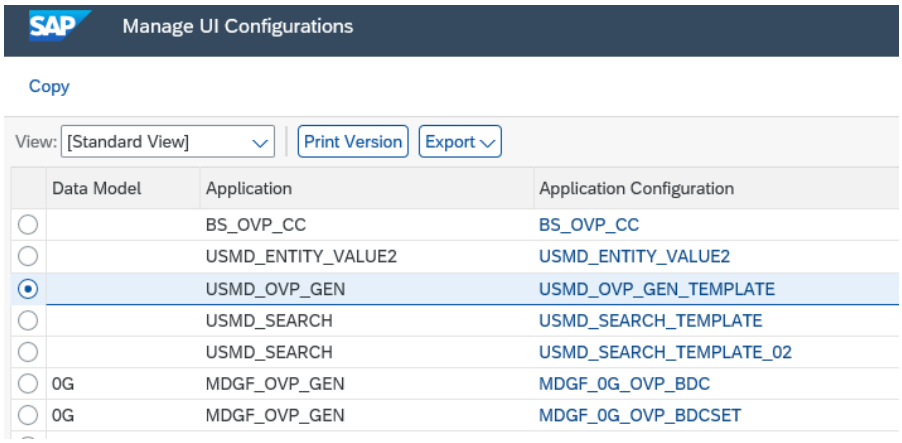
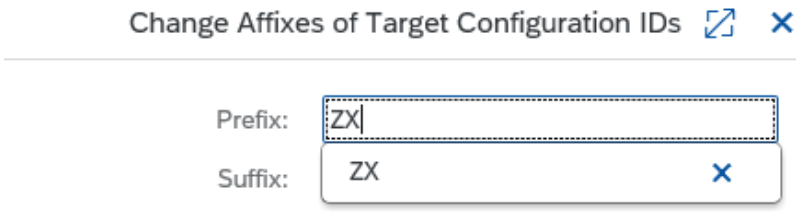
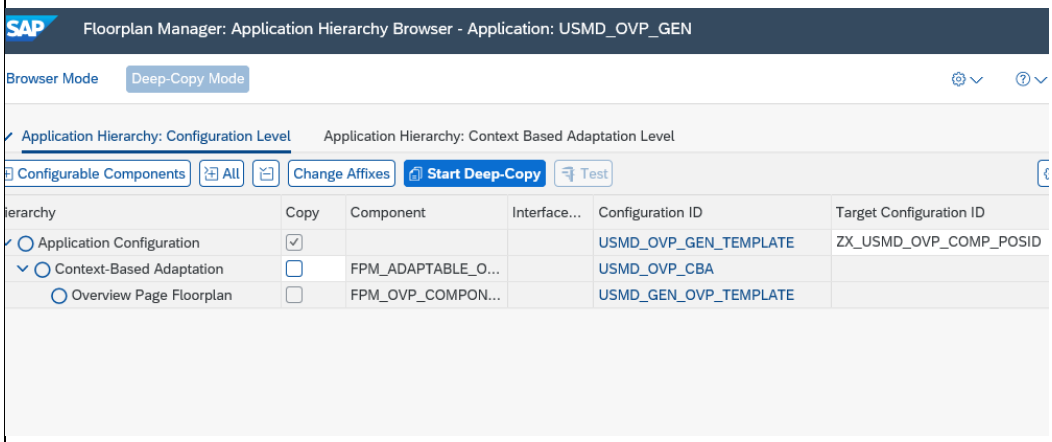
Select the pencil icon in the top left corner to edit. Then fill in the following values.	
Click Save and Exit	

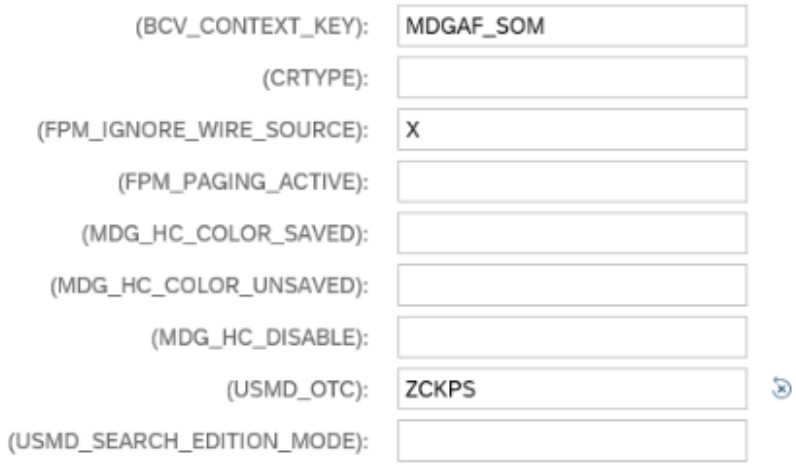
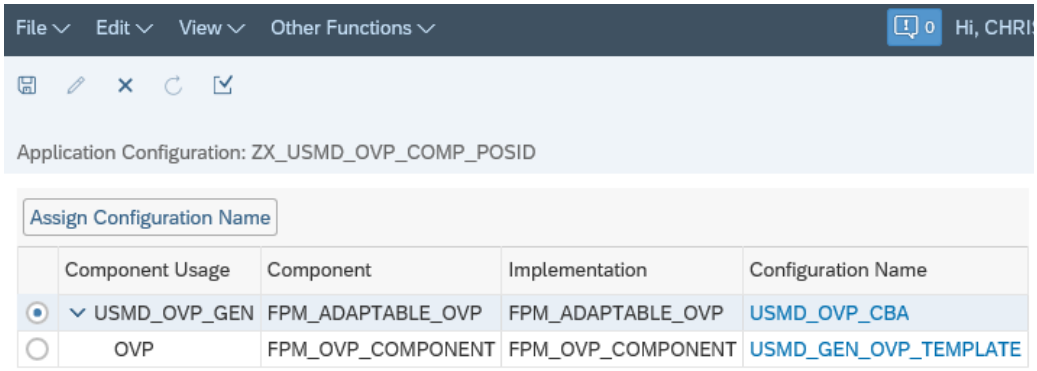
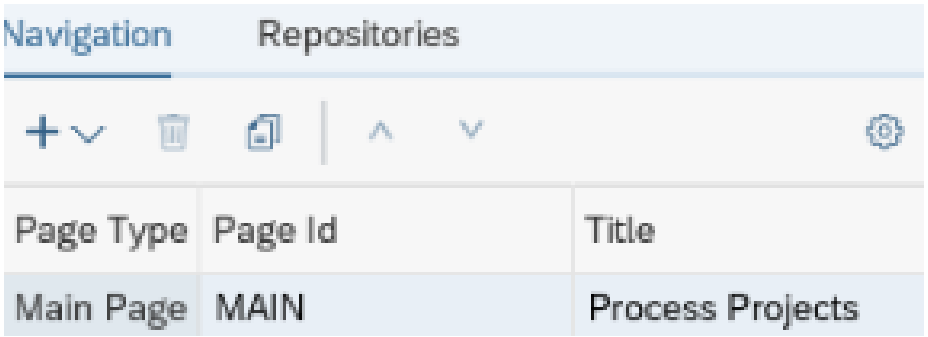
One very important step that needs to be performed after copying the generic search template configuration is to create the communicator (MDG_BS_GOV_COMMUNICATOR) configuration. The name of the communicator configuration and the search UI application must be the same; if they aren't, then the search UI application won't be rendered correctly. The communicator is responsible for rendering the search criteria and search results areas based on the search help configuration.

Now that we made changes you can see our UI configuration has our data model assigned to it. Click on details.	
Simply ensure that your component and Configuration is set as is. On the next screen, change nothing, and click **Save.**	

Develop a Single Object Maintenance User Interface

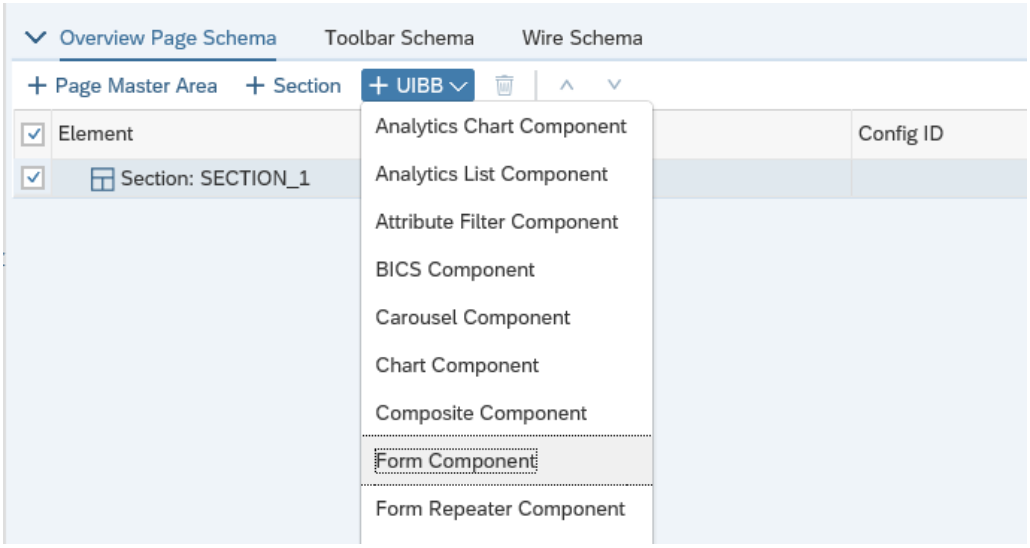
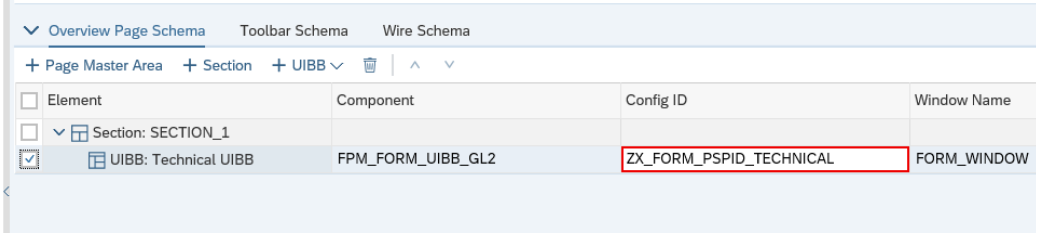
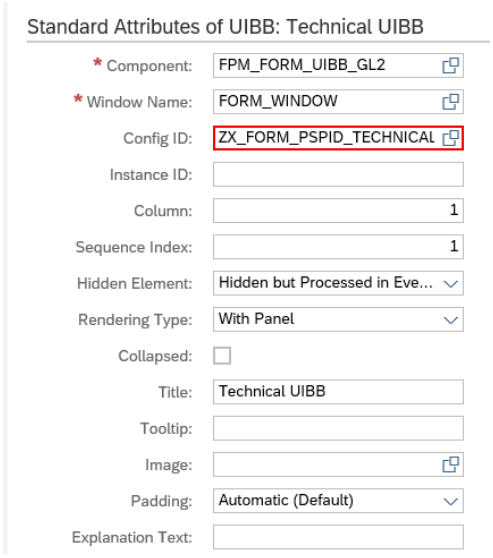
In the previous section, we created a search UI; now we'll proceed to create a Single Object Maintenance UI for the project definition and WBS hierarchy.

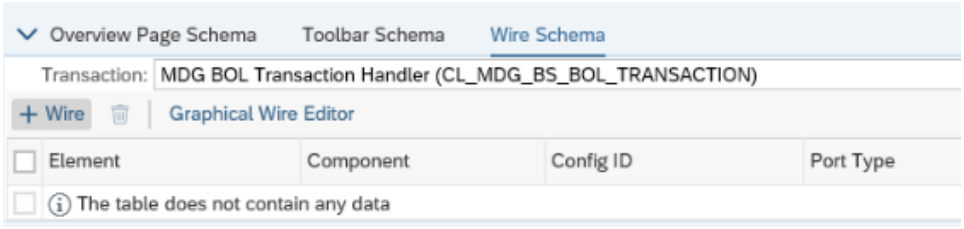
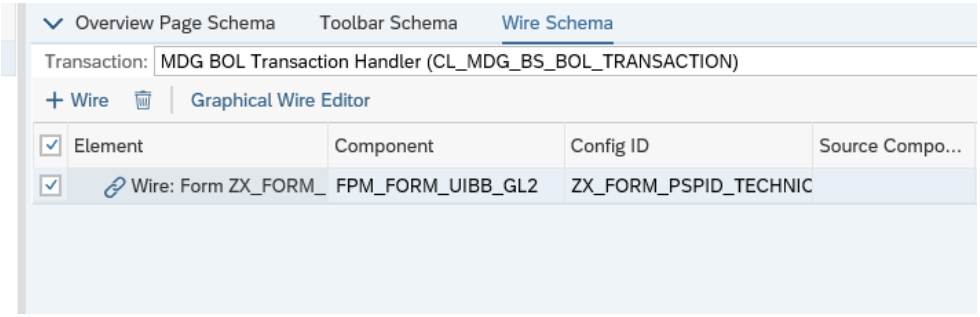
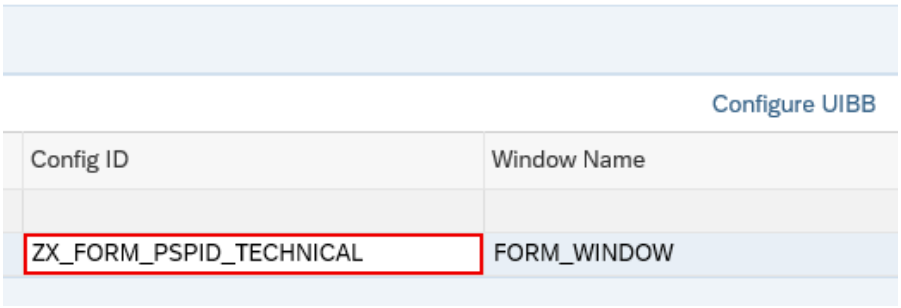
Return to the list of UI configurations and find the USMD_OVP_GEN Config. Select and click copy.	
Like before the first thing we should do is change the affixes. Like before name is as our Data Model. Select OK	
Like before rename Target configuration ID as shown. Uncheck the copy tick box for control-based application and start deep copy.	 USMD_GEN_OVP_TEMPLATE

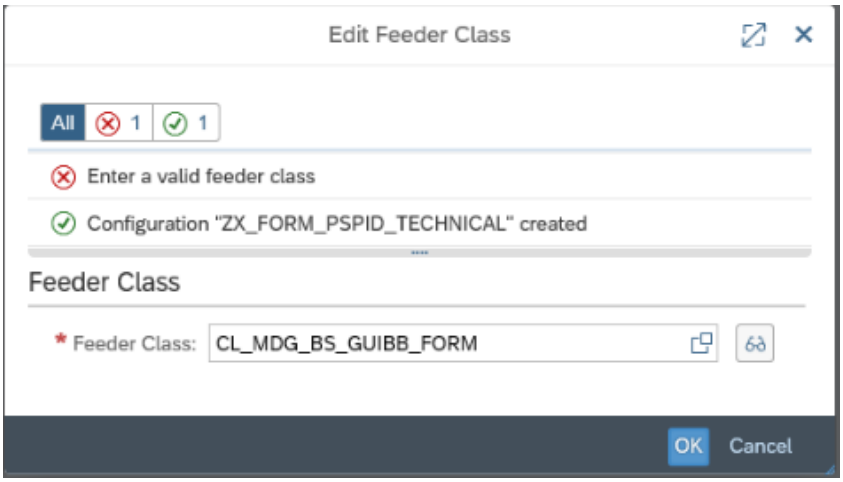
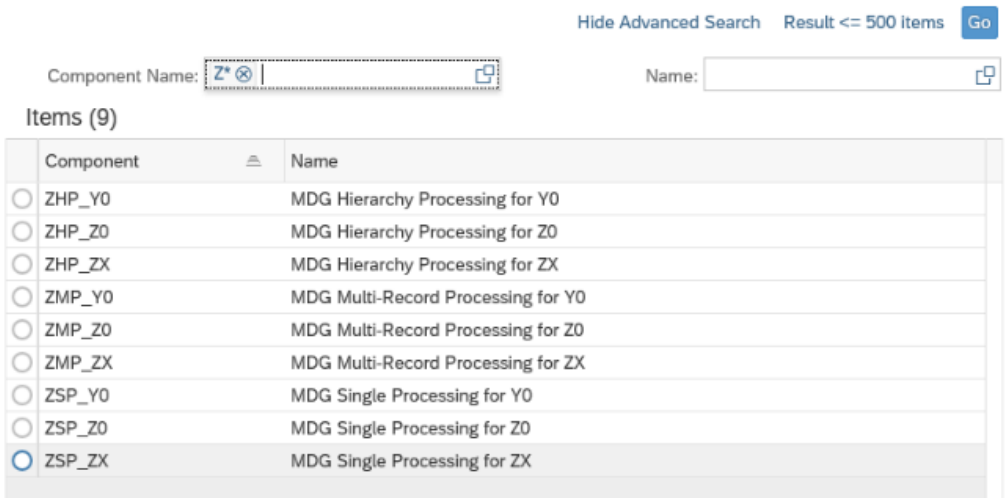
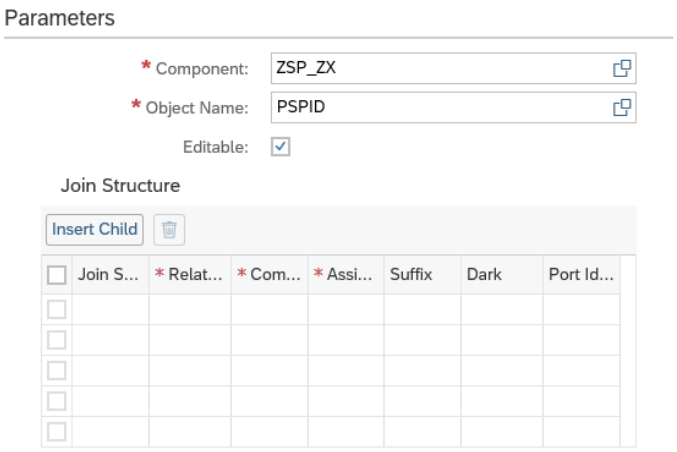
Return to the UI configurations list and select the hyperlink for your UI ZX_USMD_OVP_COMP_POSID Change the USM_OTC to suit your Business Object	
On the same page at the top we will see 'Component Usage'. Click on the configuration name USMD_GEN_OVP_TEMPLATE.	
On your new page, in the left column rename the title and page Id to a name of your choosing	

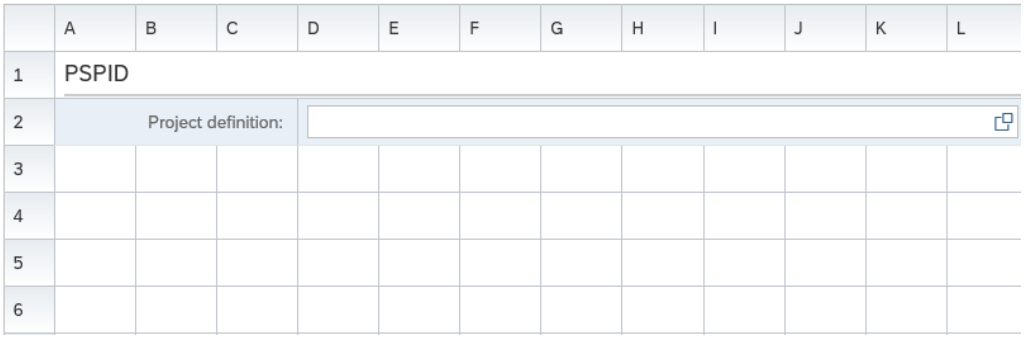
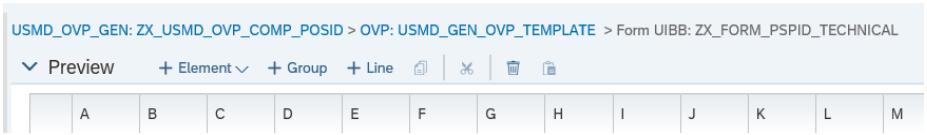
Configure the technical UIBB.

We start the UI configuration of the main page by creating a technical UIBB. The technical UIBB is never shown in the UI, but it participates in the Floorplan Manager event loop. This UIBB will be the root UIBB in the wire schema. The UIBB will just contain the key fields of entity UIBB

In the centre of your page Select the section button then select UIBB and choose Form Component.	
When opened fill in the details as shown	
Ensure you fill out the fields in the right column also. The config ID must match the name of the config ID in the previous step. We have enabled the **Hidden Element** attribute that makes this UIBB hidden but still participates in the event loop.	

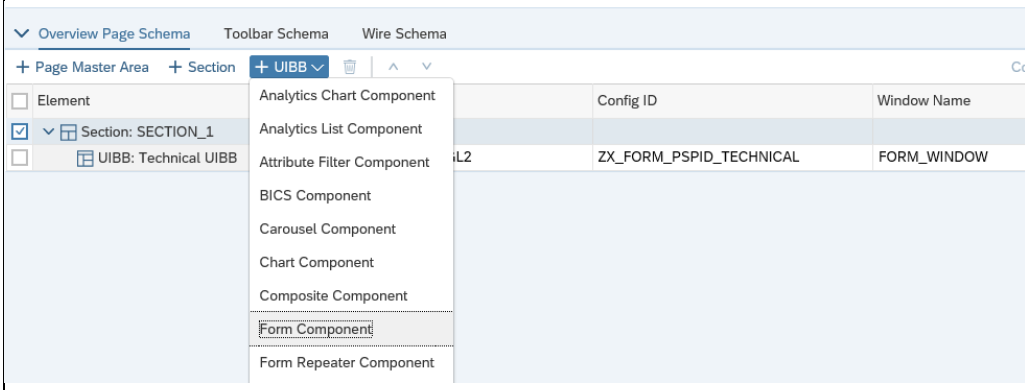
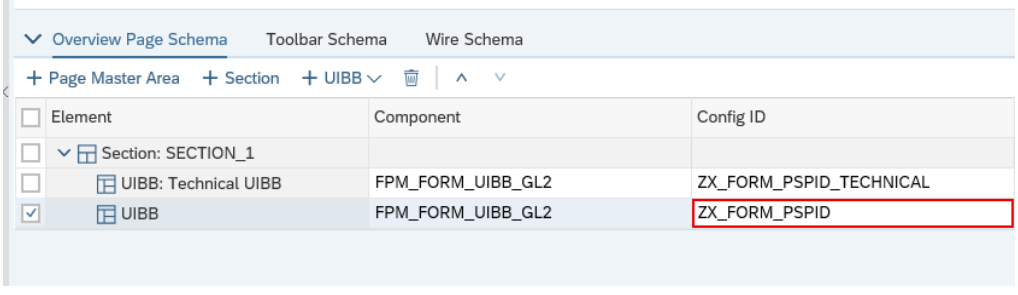
Returning to the centre column, Change the view from Page Schema to Wire Schema. Here we will create a wire, which has the technical UIBB as the target UIBB but doesn't have a source UIBB. This makes it the root UIBB, which can be instantiated independently. Select the + Wire Button	
The component should automatically be filled, it is important you fill out the cofig ID with the same name in the previous steps.	
Save your changes	
Return to the Page Schema view and select configure UIBB in the top right corner.	

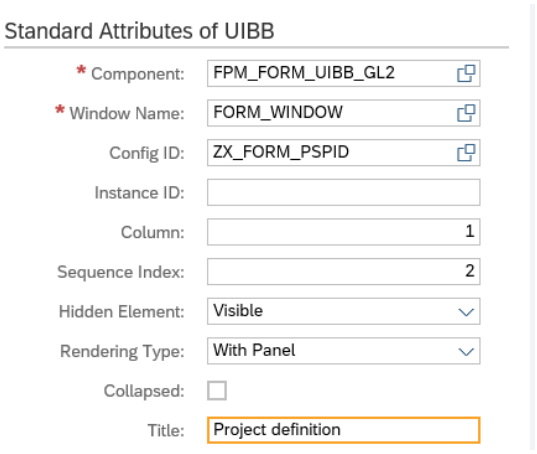
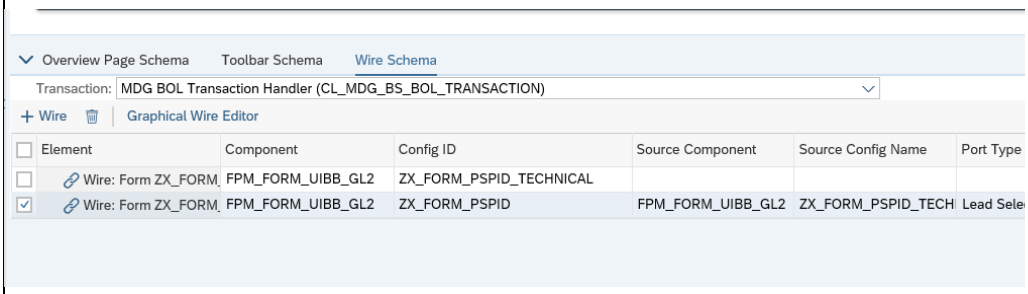
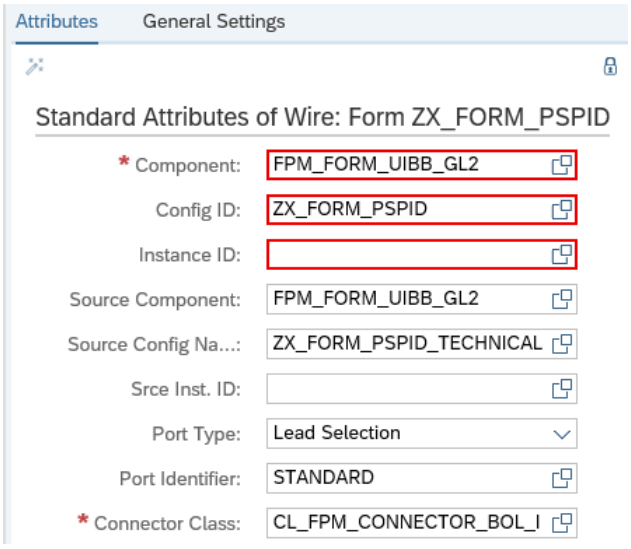
When you come to the feeder screen. Insert the feeder class as shown and click ok	
Open the search in the component parameter and select 'ZSP_ZX' as shown	
Insert PSPID as the object name and it is important to always tick the editable box.	

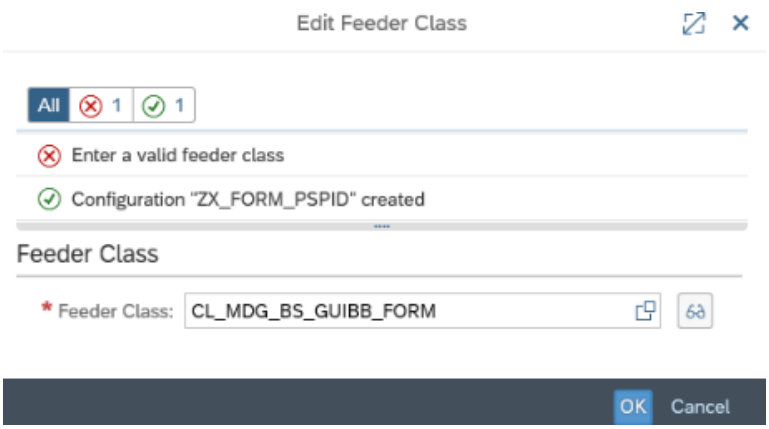
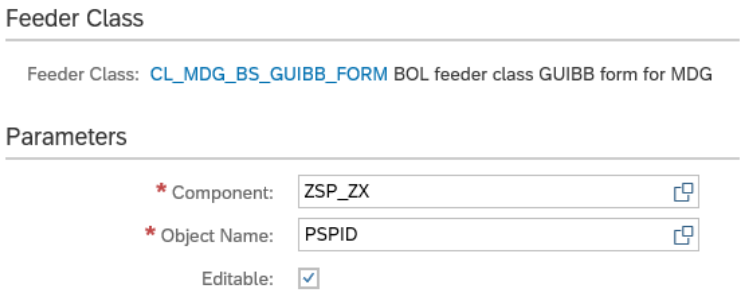
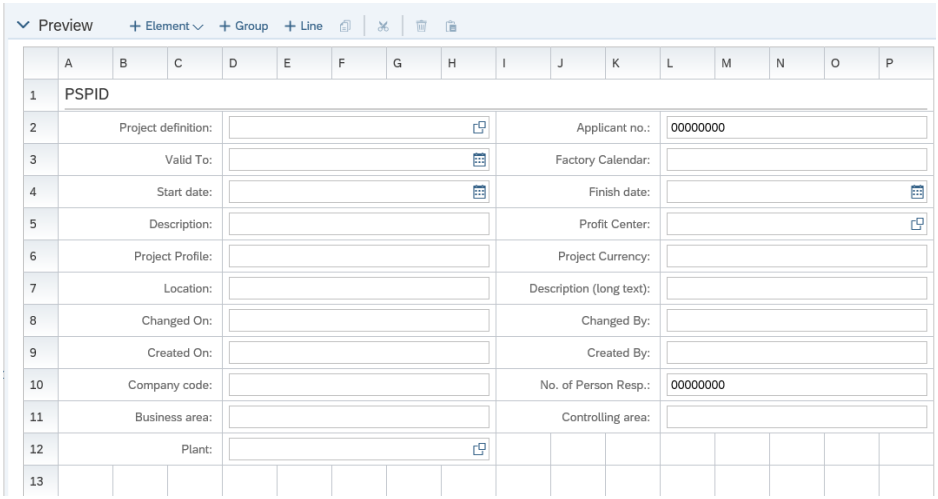
On the new screen ensure that Project Definition is the only element. You can remove everything else.	
Make sure to save. Return by selecting the hyperlink as before: *USMD_GEN_OVP_TEMPLATE*	

Configure form UIBB for project definition.

Here we will add the first visible UIBB. This UIBB will be a form and will hold the project definition. We will also configure a wire with port type lead from the technical UIBB to the project definition UIBB.

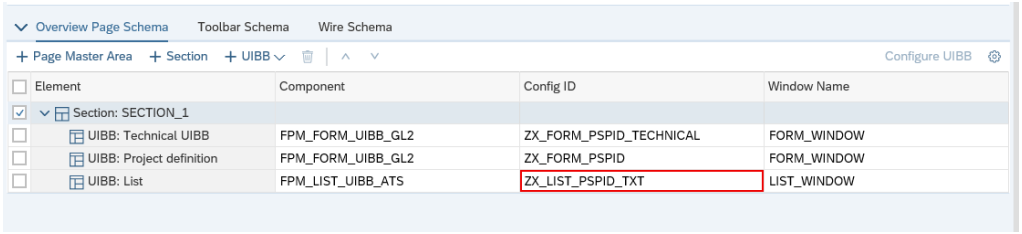
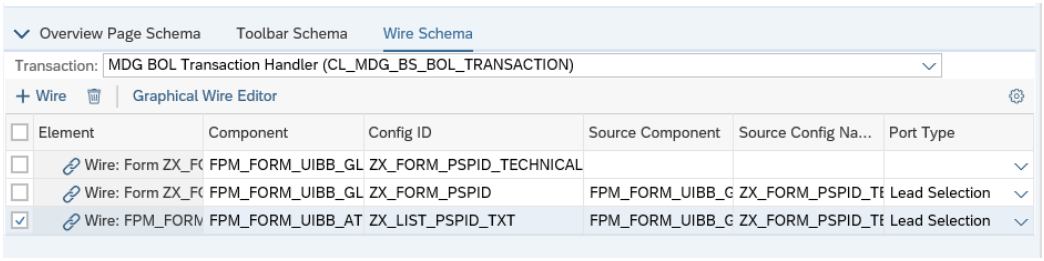
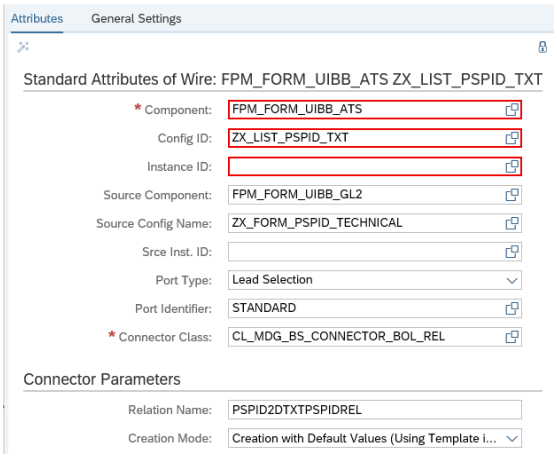
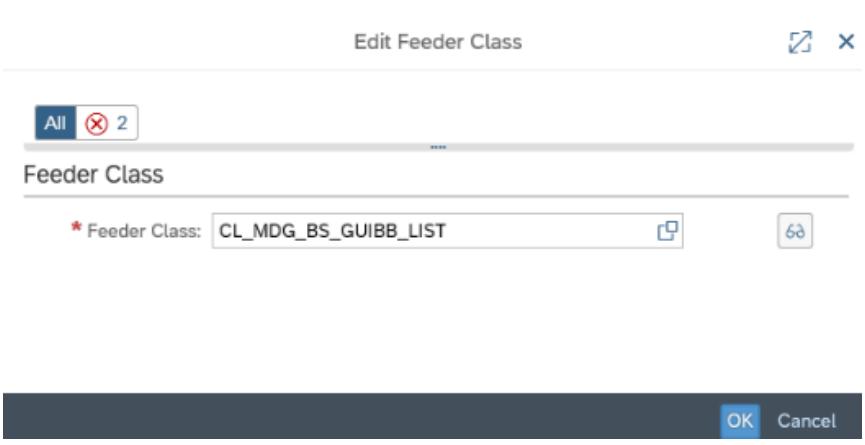
Create a new UIBB screen as before. Again, create a Form Component UIBB	
Like before fill in a configuration name.	

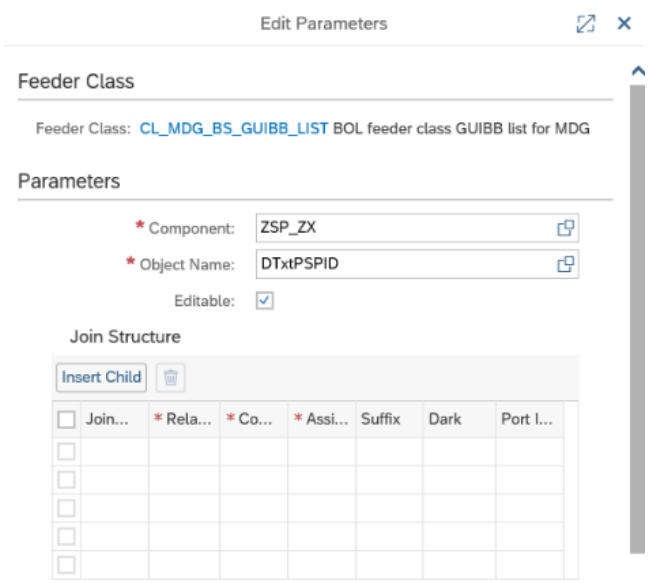
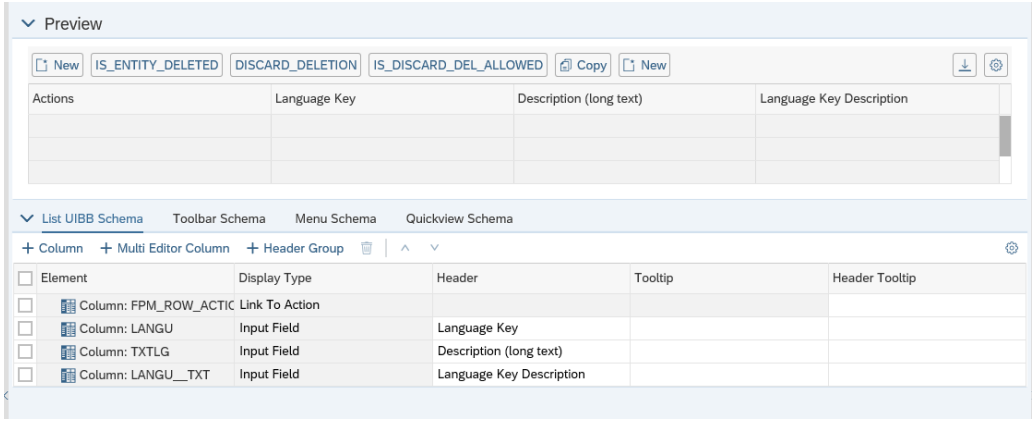
Entering the details in the right column ensure we write the configuration name the same. Note this time we have chosen this as visible	
Like before Create a wire and fill out the config ID same as last step. The source config name will be the name of the first wire we created *ZX_FORM_PSPID_TECHNICAL* Change port to lead Selection	
In the right column ensure the details are filled out as the same. Input the port Identifier and Connector class: *CL_FPM_CONNECTOR_BOL_IDENTITY*	
Save Changes	

Like before select the Configure UIBB. We will be using the same Feeder class in this case.	
Input the details the same as the previous UIBB.	
This time we can have more elements. Arrange your elements as shown in the picture.	
Save and return as previous	

Configure the list UIBB for language-dependent texts

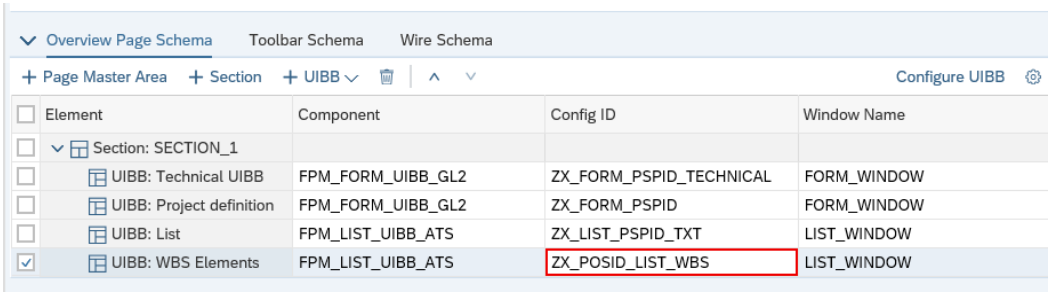
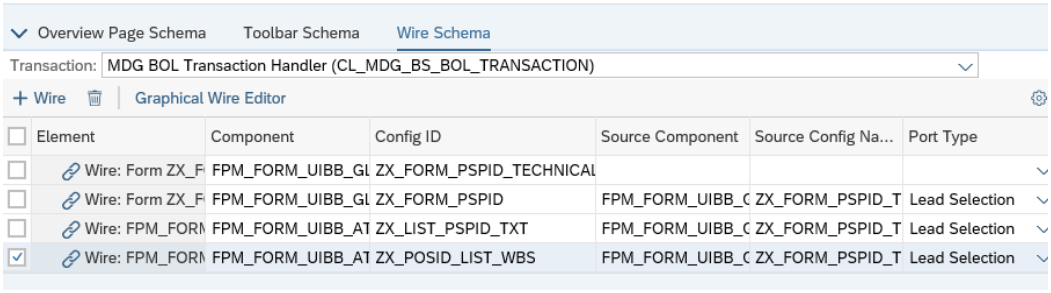
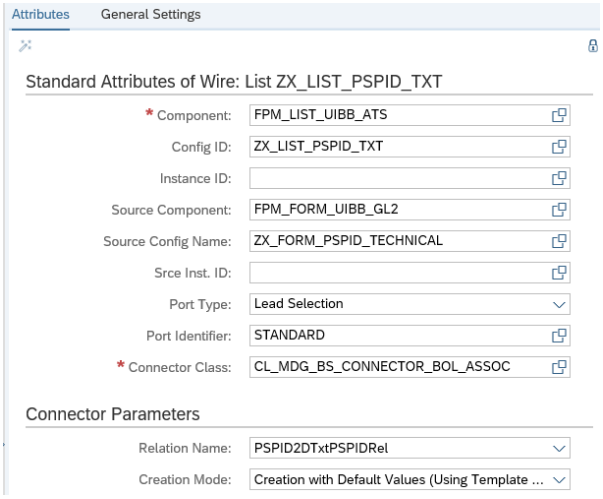
The project definition (PSPID) has language-dependent texts activated, allowing the description for the project definition to be maintained in multiple languages. To enable this functionality in the UI, you need to add a list UIBB.

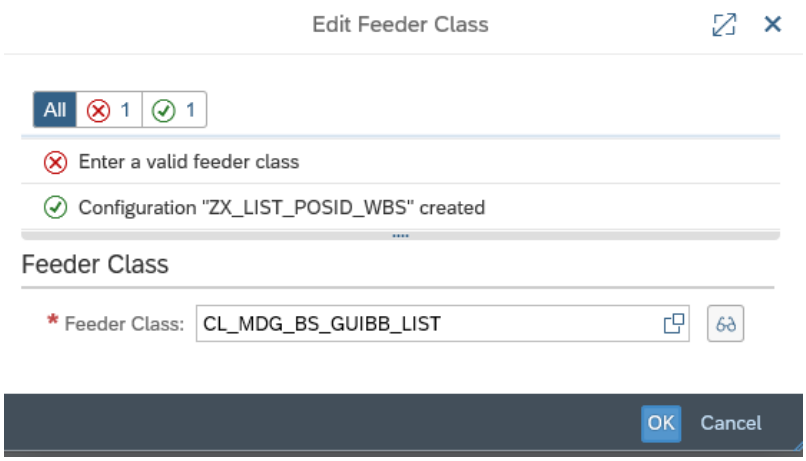
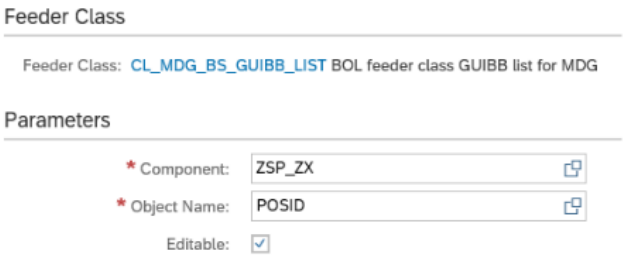
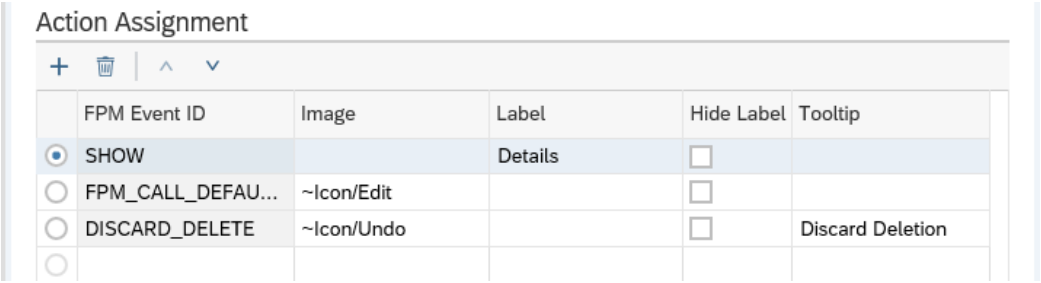
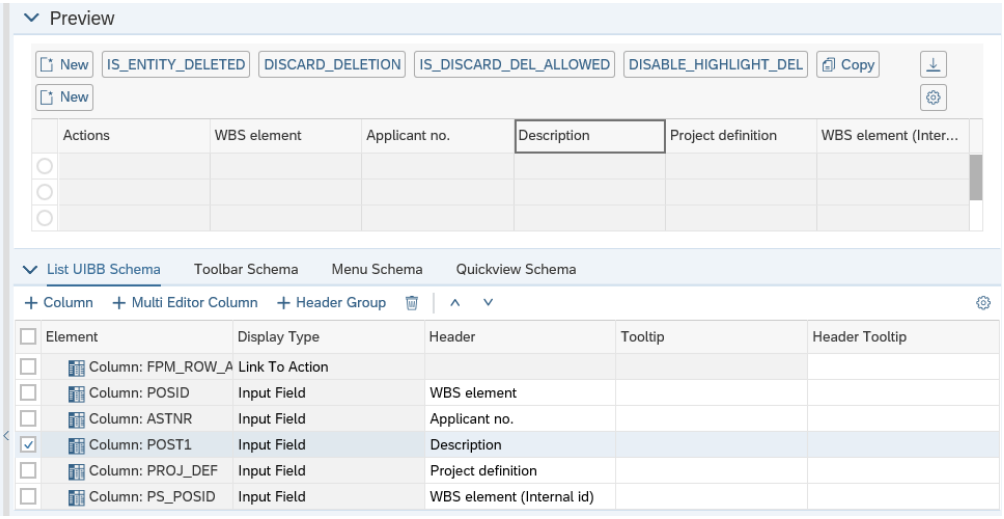
Create a new UIBB as previous except this time select List component instead of Form. Fill out the config ID and ensure the details in the right column are filled out correctly.	
Create a new wire as previous. Insert a Config ID and name the other columns as previously done.	
In the right column fill out the details as shown. Note we are using a different connector class this time. Ake sure to fill out the connector parameters as shown	
As previous select configure UIBB. This time we are using a different feeder class. Insert the feeder class shown	

Again select the component as ZSP_ZX. Insert the object name as DTxPSPID	
Arrange the elements as shown in the picture. Any additional elements can be removed	
Save and return as previous	

Configure the list UIBB for creating and changing WBS elements

A project definition can have multiple WBS elements assigned to it. Each WBS element has its own set of attributes and is assigned to the WBS hierarchy. To meet these requirements, we first create a list UIBB in the main page and then create an edit page; this edit page is tagged as the default edit page for the list UIBB.

Like the previous create a List Component UIBB. Name as shown and ensure the right column is filled out correctly	
Create a wire like previous with the same Source component and Source Config name	
In the right column fill out the details as shown. Note we have changed the Connector class again. The port identifier is the same.	
Save and click configure UIBB	

Insert the feeder class, we are using the same feeder class as previous	
Fill out the details as shown. Select OK	
In the right Colum you will see an 'Action Assignment '. Insert Events: *Show* *FPM_CALL_DEFAULT_EDIT_PAGE*	
Arrange your UIBB layout so it appears as shown in the picture.	
Save Changes	

We have now fully implemented the UI for our custom data model.

Process Modelling

In MDG every change including the creation of master data is done through a change request, which is roughly a carrier of changed to the master data. After change request are initiated, they need to be processed by applying governance rules and collaboration.

The process model provides the required input for change request creation and process of change requests. The process modelling configuration node in SAP MDG is a group of similar configuration activities required to execute the change request process.

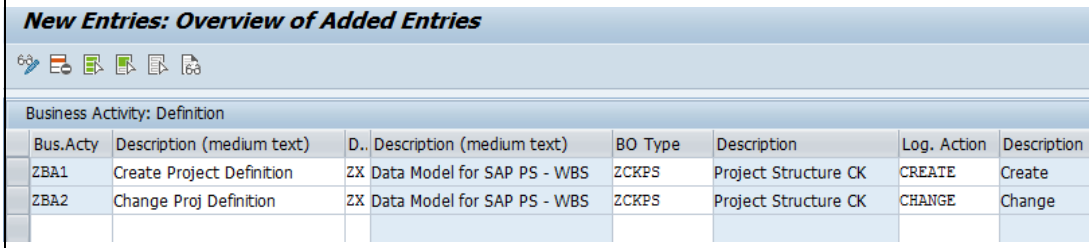
For process modelling, the first step is to create a new business activity to tie an action with the business object and then maintain navigation settings of the custom UI application using business activities and actions.

Create a New Business Activity

To design a business activity, you first must understand what kind of actions your intent to perform and on which business object. Each combination of logical action and business object becomes one business activity. In our case, we will create two busines activities.

ZBA1 Create project definition and WBS elements.

ZBA2 Change project definition and assign WBS elements to the project hierarchy.

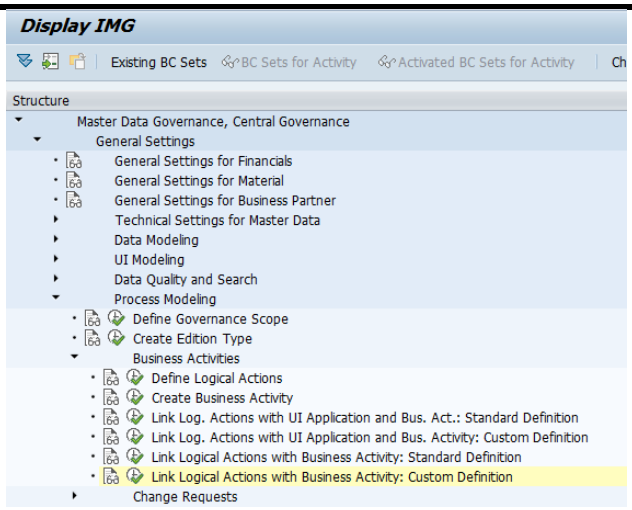


Assign Business Activities and Logical Actions to Business Objects

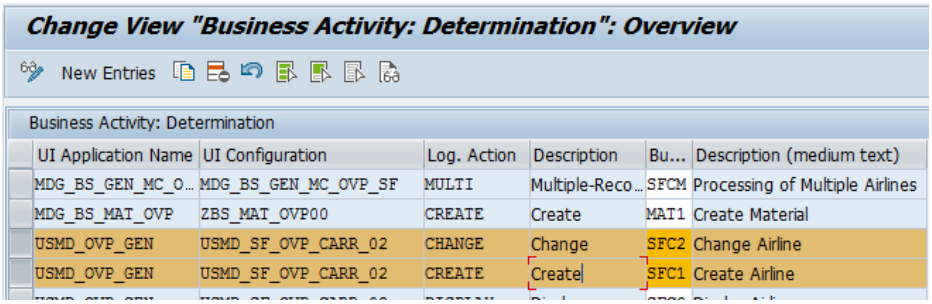
A logical action represents the operation to be performed on the master data by an actor in the process (e.g., create, change, or delete). Business activities add business context to logical actions by linking them with business objects such as create supplier, change material, and delete account. Business activities are defined by assigning an action, data model, and business object.

In transaction mdgimg:

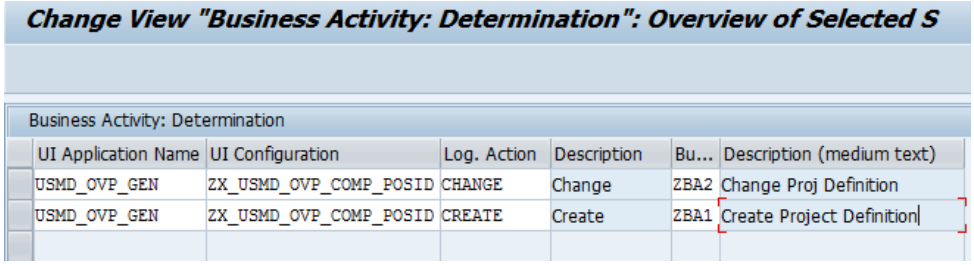
General settings → Process Modelling → Business Activities → Link Logical Actions with Business Activity Custom Definition.

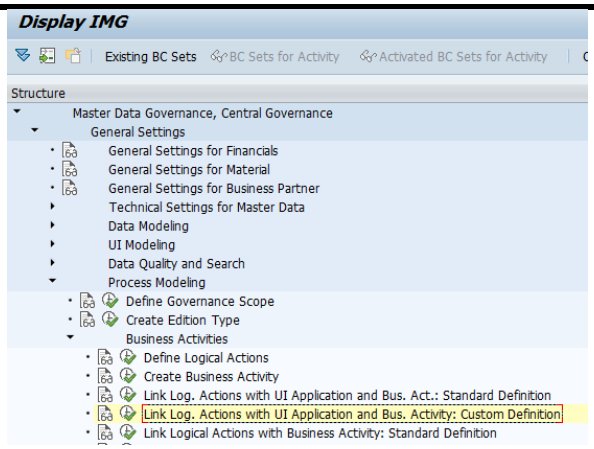
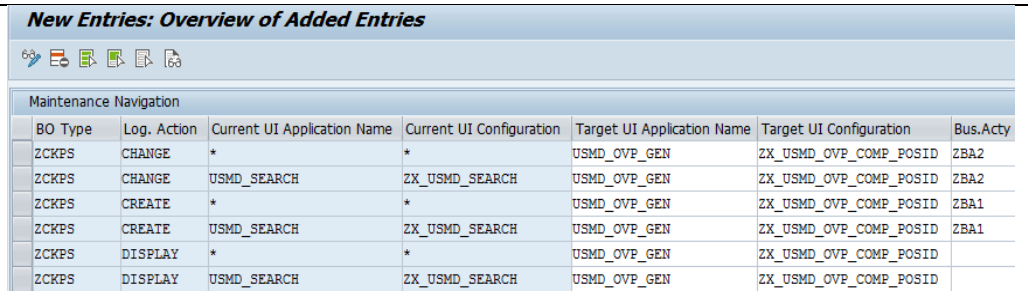


In our case we don't need to create a new action but reuse a SAP delivered action such as change and create. To do this select the two actions with change and create and click copy as



We can no edit those actions to suit out UI Configuration and Business activity as shown.

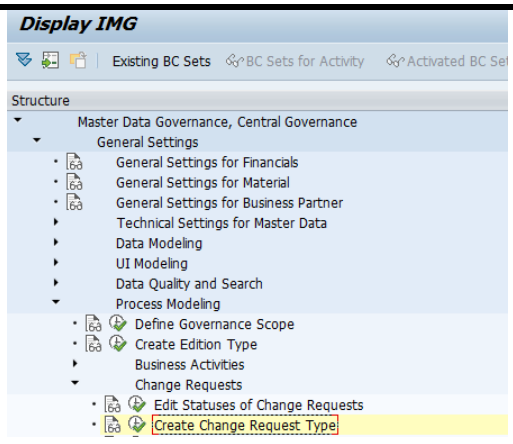


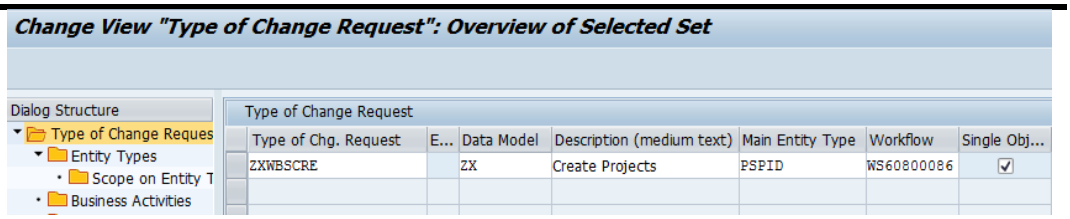
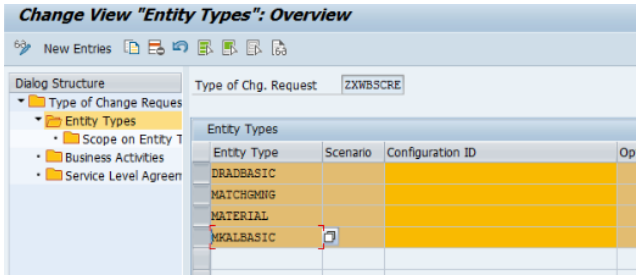
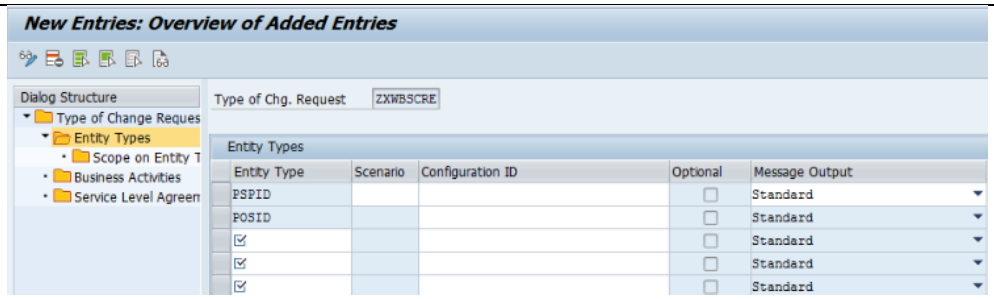
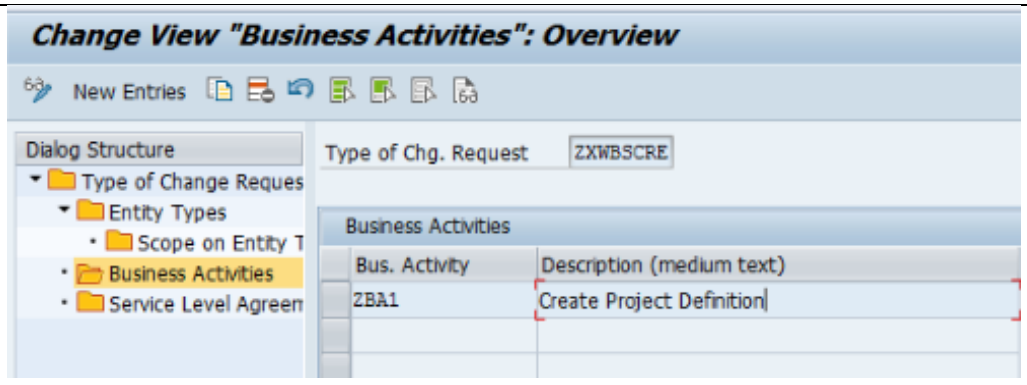
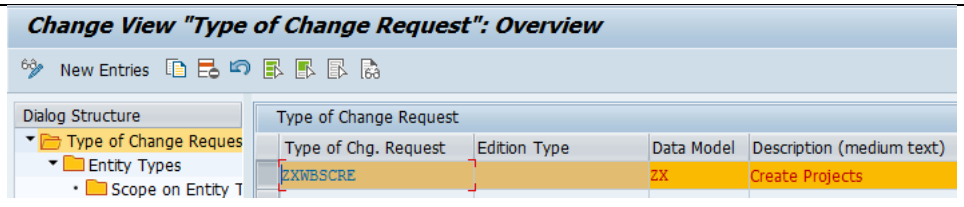
Creating cross- application navigation. Firstly like before access this through mdgimg General settings → Process Modelling → Business Activities → Link Log Actions with UI Application and Bus. Activity Custom Definition.	
Select new entries and enter the values shown. This is the list of configurations created for cross-application navigation between the generic search UI and the custom object UI for actions create, change, and display.	

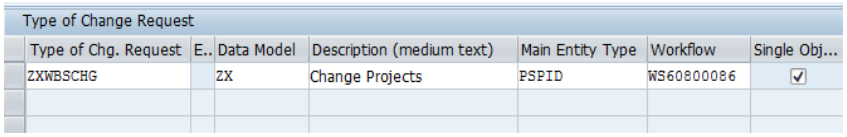
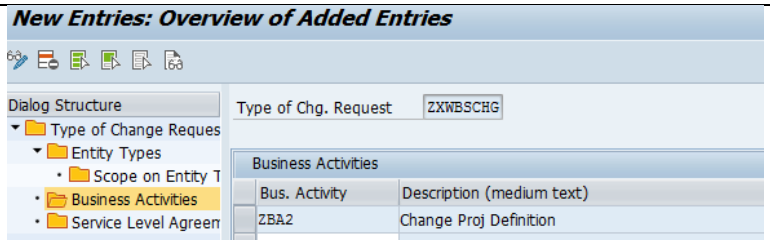
BO Type	Log. Action	Current UI Application Name	Current UI Configuration	Target UI Application Name	Target UI Configuration	Bus.Acty
ZCKPS	CHANGE	*	*	USMD_OVP_GEN	ZX_USMD_OVP_COMP_POSID	ZBA2
ZCKPS	CHANGE	USMD_SEARCH	ZX_USMD_SEARCH	USMD_OVP_GEN	ZX_USMD_OVP_COMP_POSID	ZBA2
ZCKPS	CREATE	*	*	USMD_OVP_GEN	ZX_USMD_OVP_COMP_POSID	ZBA1
ZCKPS	CREATE	USMD_SEARCH	ZX_USMD_SEARCH	USMD_OVP_GEN	ZX_USMD_OVP_COMP_POSID	ZBA1
ZCKPS	DISPLAY	*	*	USMD_OVP_GEN	ZX_USMD_OVP_COMP_POSID	
ZCKPS	DISPLAY	USMD_SEARCH	ZX_USMD_SEARCH	USMD_OVP_GEN	ZX_USMD_OVP_COMP_POSID	

Create Change Request Type

The change request type is a key characteristic of the change request that determines how a change request is processed. The change request type links a change request to the workflow, data model and business Activity.

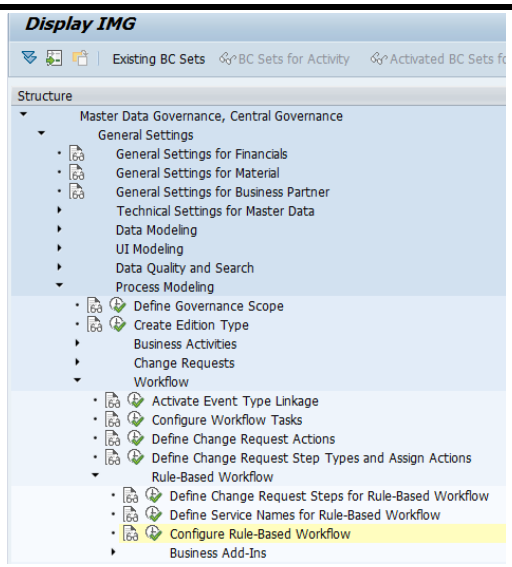
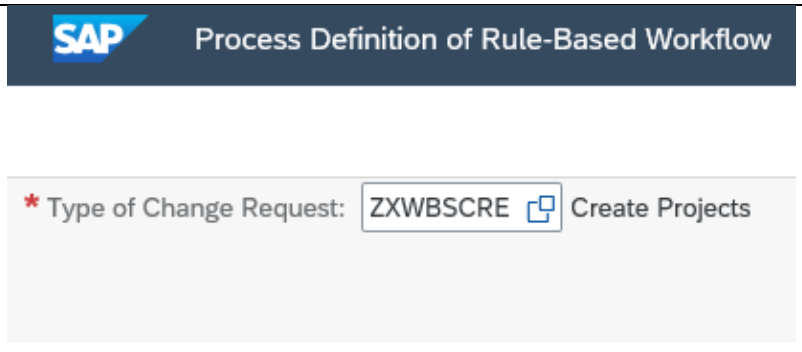
Open Create Change request type through the mdgimg transaction as shown	

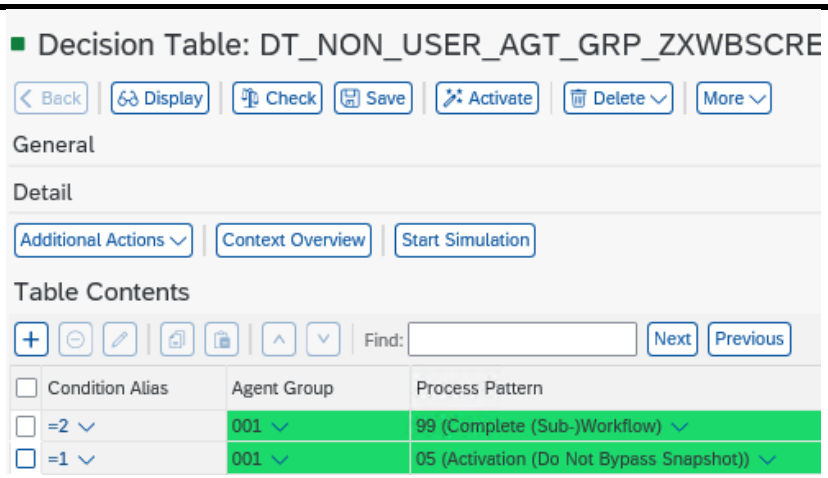
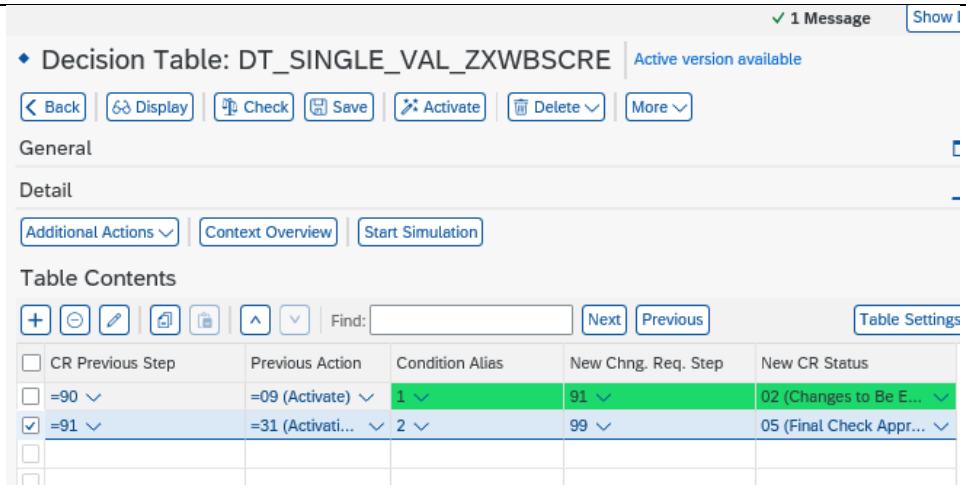
Find the Mat01 Change Request and select copy. Then change the details to as shown	
We will need to delete the Entity Types select all and then remove.	
The select new entries and add the two Entity Types shown.	
Open business activities and like before remove the activity associated with MAT01 and then add the business activity we created for create.	
We need to create a 2nd for change. We can simply copy the Change request type we just created.	

Change the details to as shown to associate it with changing projects.	
Change the business activity to the one we created for change Projects	
Save your Changes	

Create a workflow

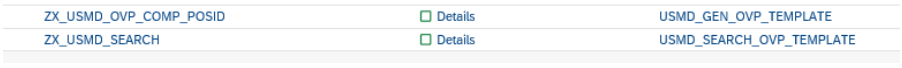
We need to assign a workflow to the change request type In our case we will assign a rule based workflow as it is more structured.

Like previously we will be opening this through the mdgimg tab. Open configure rule based workflow as shown in the picture.	
Insert the create change request type we made as shown	

Locate the Decision tables and Open the Non-User decision table. Insert two rows as shown.	
Save and Activate	
Next open the Single Value Decision table and fill in the values as shown.	
Save and Activate	

Creating our CR Wire

Now that we have created our Change Request and assigned a workflow, we must create a link between that and our UI. We have created our UI and our CR although as of now they are not linked and our model wont work. We need to create a CR wire to do this in our UI configurations.

Locate the UI configurations page as we did earlier and find our UI configuration. Select Details Then select change	

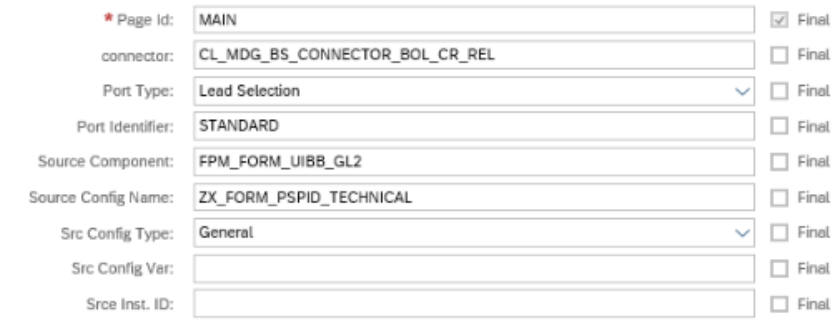
We need to create a wire to connect our change request with our UI.

Select new → crWires

Fill in the details as shown.

As you can see the page Id represents the page we created earlier.

The source config name is our Technical UIBB.



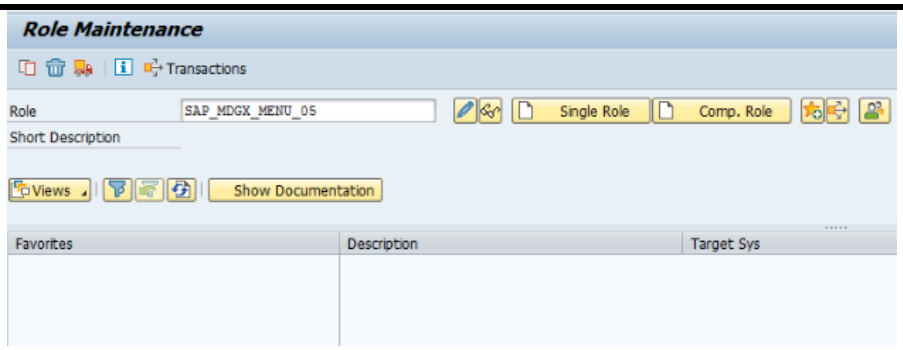
Creating our Role

Roles are used for configuring authorization profiles and menus for users. Roles are directly assigned to the user master and can also be used to configure personalization values for users. The role for menu is used exclusively to configure the menu entries in SAP Business Client.

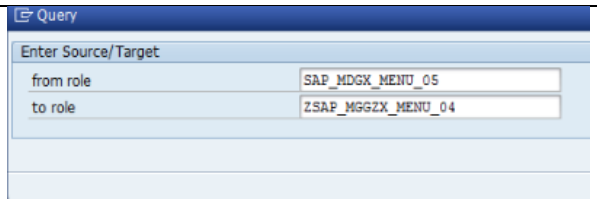
Search Transaction 'PFCG'.

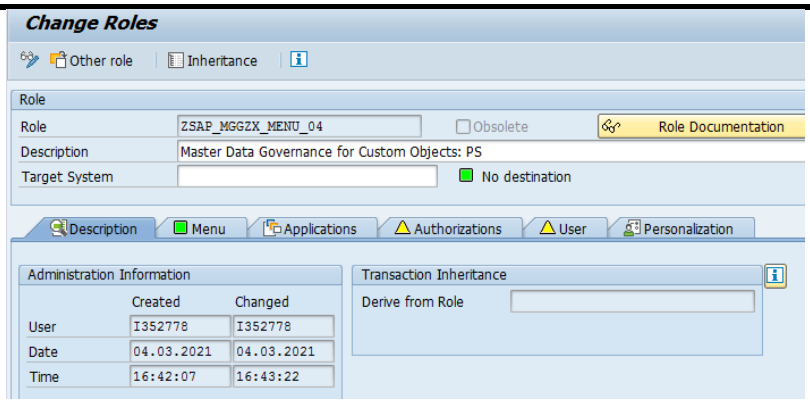
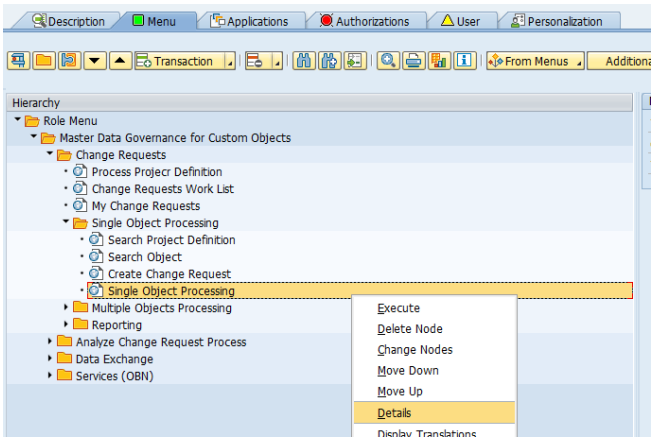
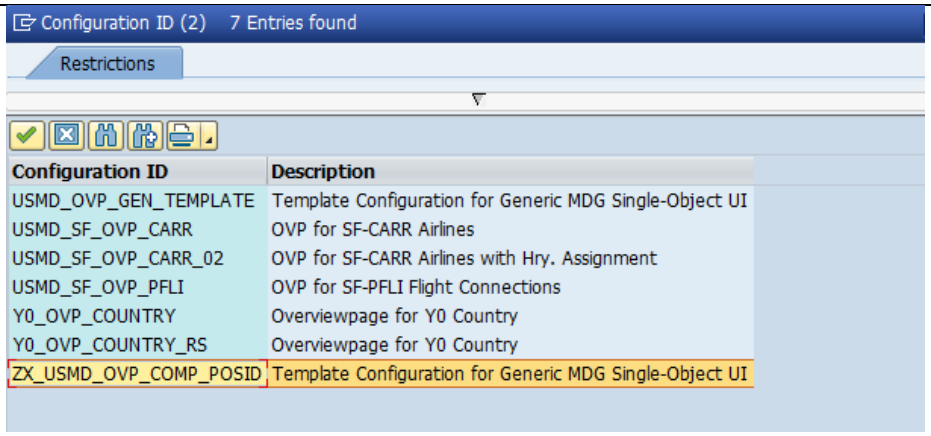
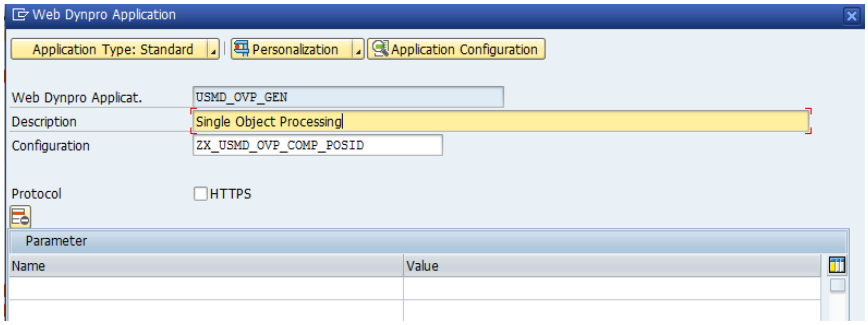
Enter the role 'SAP_MDGX_MENU_05'

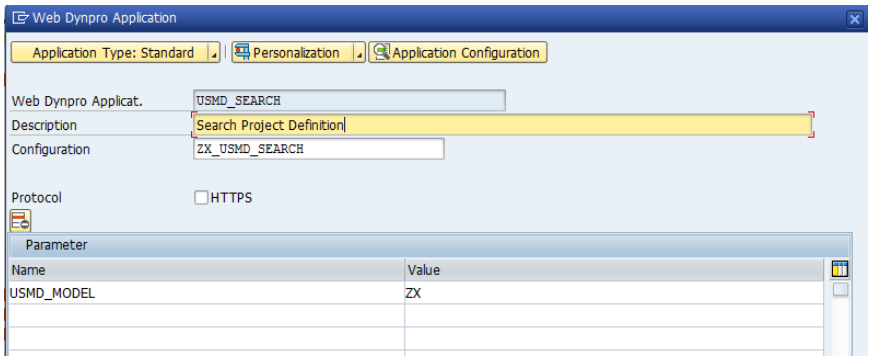
Then select copy in the top left corner.



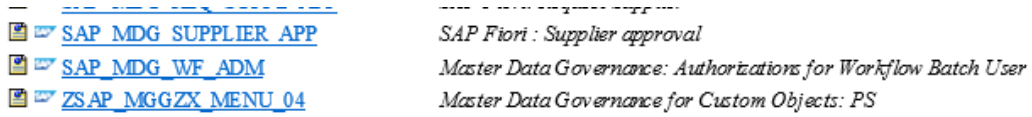
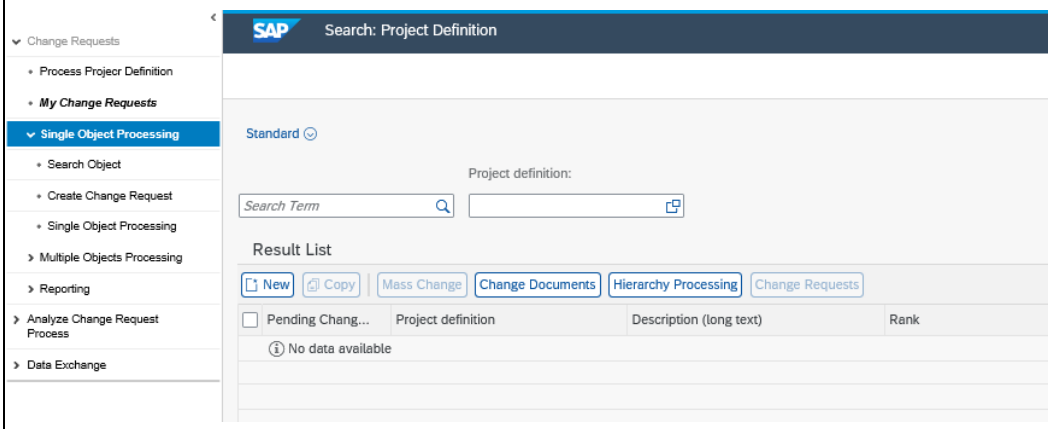
Then enter the name of your role and select ok



Enter a description for your role	
Save	
Select the Menu tab and then open single Object Processing The right click on the single object process tab and Select details.	
Enter the Configuration Id as the ID we created. You can search this to ensure you select the correct one.	
You can change the description or leave as is then click ok	

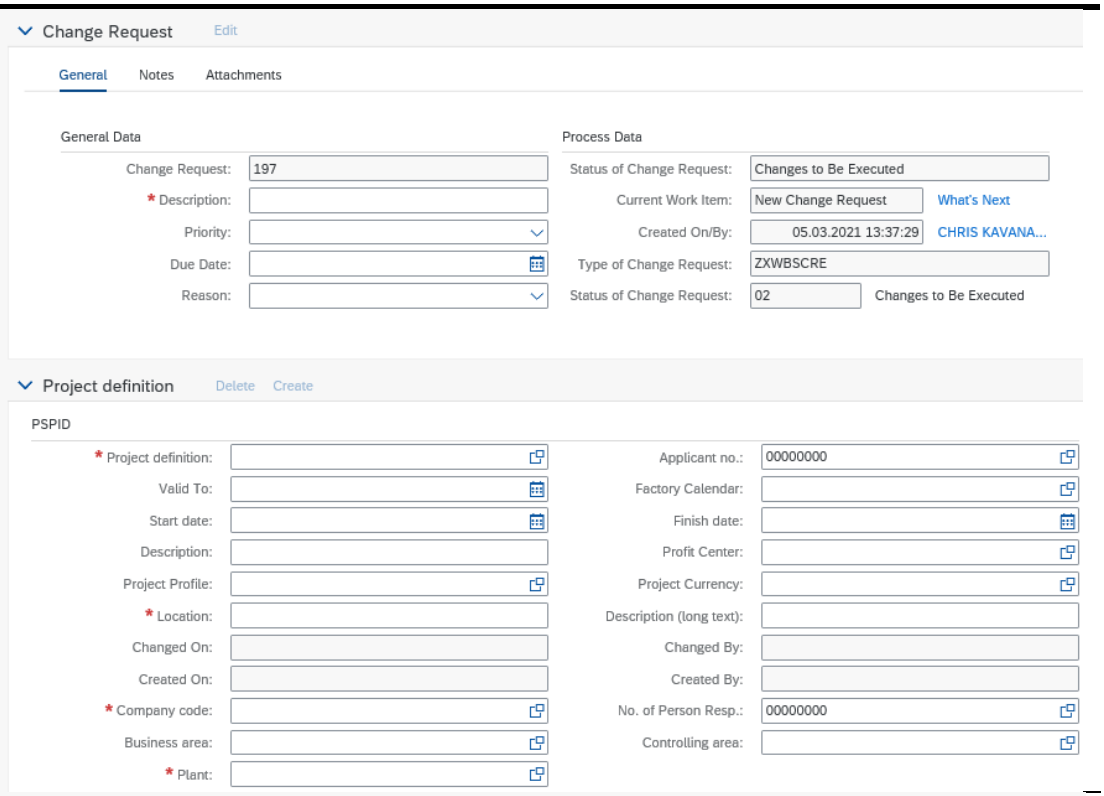
Like before open the Single Object Processing folder, then right click on the Search Object and click details. Change the configuration to the Search config you created. You can change your Description or leave as is. Input your Model as shown.	
Save your role.	

Testing our Data Model

Open Transaction NWBC. Scroll down and select the role we created	
Select Single Object processing as you can see we will be brought to our Search screen. Select new to create a new PS	

Next we will see the whole UI we created.

The Project definition attributes.



We can additionally see WBS elements option.