ODATA

What is OData and REST in SAP Netweaver Gateway

- SAP Netweaver Gateway provides REST based and open interface that implements simple access to SAP systems via OData protocol.
- With SAP Netweaver Gateway any one can i.e. any user interface(UI) can access the data through OData protocol.
- Is this the only one that can communicate to other systems in SAP portfolio? the answer in NO. We have SAP PI system to exchange information between different systems and especially between a company and 3rd parties.
- But the main difference between SAP PI and SAP Netweaver Gateway is in their usage.
- SAP Netweaver PI is to communicate the information between system-to-system, but SAP Netweaver Gateway is system-to-user.

What is OData?

- OData is REST based protocol to access and update the data. OData is build on Atom Publishing Protocol(AtomPub), XML and JSON. This make the protocol easy to understand and use.
- In short OData is an Online Database Connectivity(ODBC) for the web. It offers a simple and uniform way of sharing the data on protocol level which in turn enables broad integration across different products and platforms.

What is REST?

- OData is REST-inspired technology for reading, writing, and modifying information on the Web (not just SAP). **REST** = **RE**presenational **S**tate **T**ransfer. *REST is an **architectural style** that uses simple and lightweight mechanism for inter-machine communication*. It is an alternative to the RPC (Remote Procedure Calls) and Web Services. REST is ***Resource-based***, unlike RPC or SOAP which are ***Action-based***.
- REST services are called as REST services because the Services are really working with ***Resources*** instead of ***Operations***. Any communication between client and services are using ***URI*** (Unified Resource Identifier) over HTTP protocol using HTTP method. The URI is really the representation of the Resources (like POHeader, POItem, Customer, Vendor etc). Also, in RESTful service, once you identified the Resource, you will be working with a uniform interface, because it uses HTTP methods (GET, PUT, POST and DELETE) to work with the resource. *So, the client does not need to know what the exact operation name defined in the service contract to call that method.* GET method is used whenever we need to get the representation of an existing resource. POST is used to add new resource into the system. PUT is to modify the existing resource and DELETE is to remove the resource from the system. *No matter what is the Service in whatever Platform, GET, PUT, POST, DELETE remains the same.*

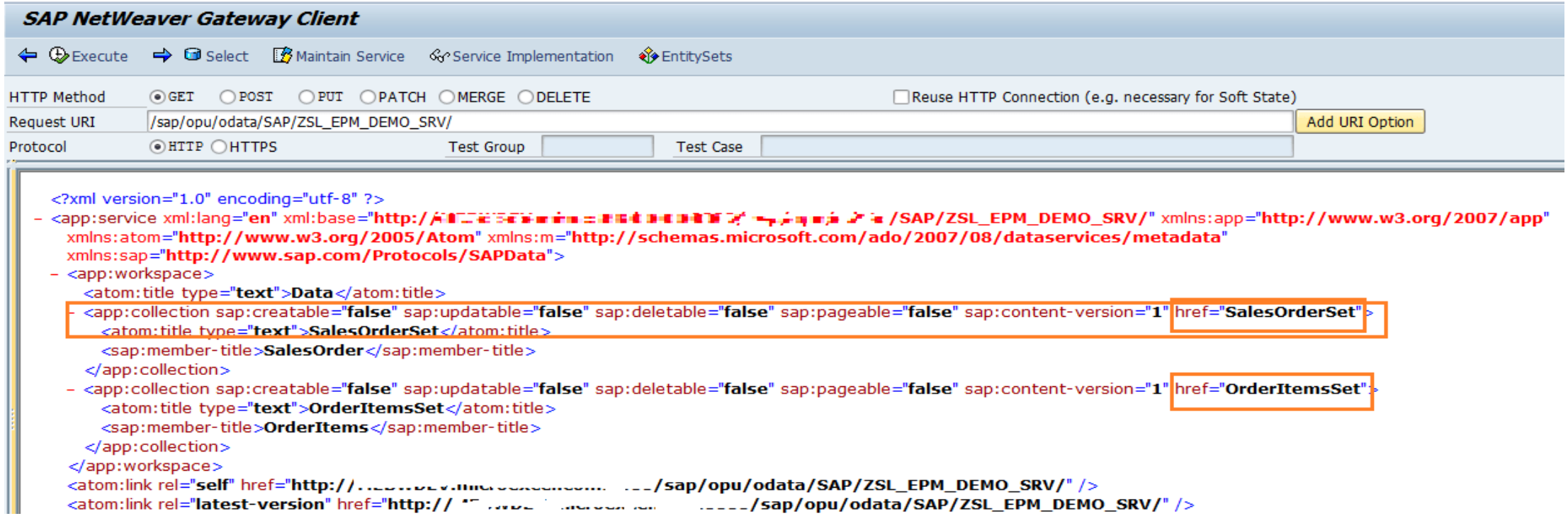
Structure of OData Service

It have mainly two parts

1. Service document
2. Service metadata document.

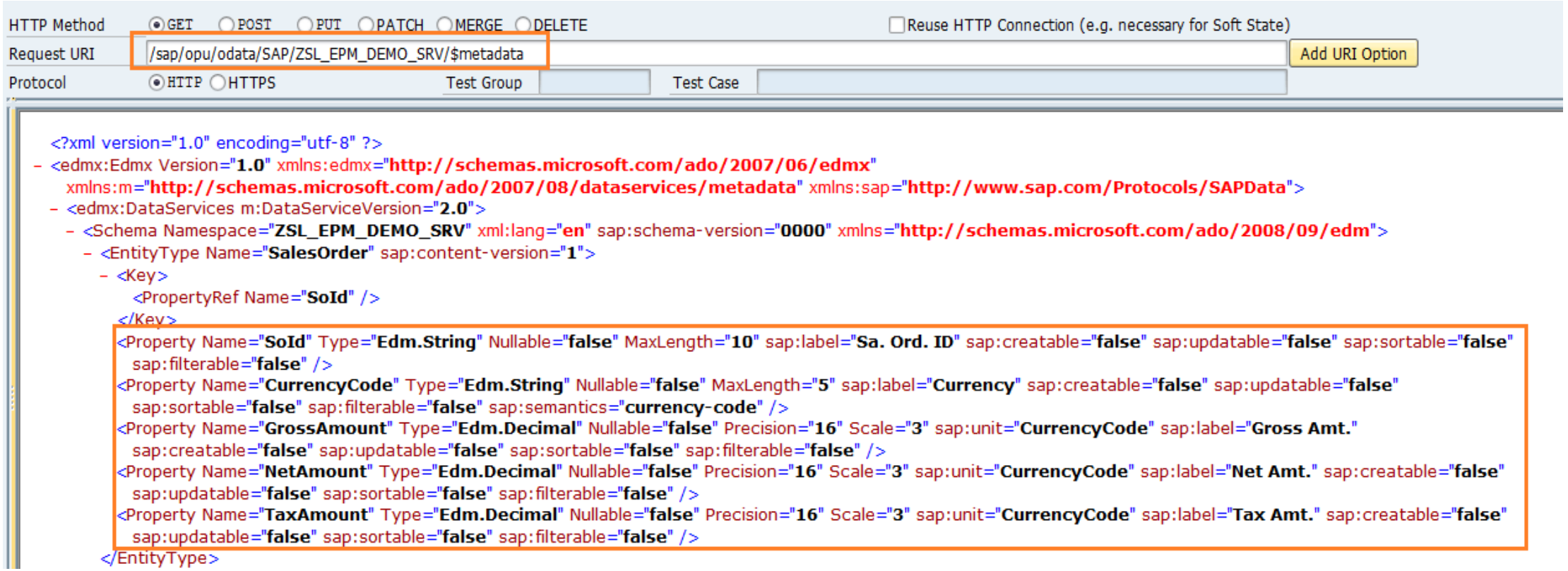
Service document

- Service documents consists of all list of resources URI's that can be accessible. ZSL_EPM_DEMO is the OData service which we will be going to build in future tutorials. Basically this service will retrieve sales order data.
- Lets look at what information does this service documents holds.Service document is accessible through the URI.



Service metadata document

- Service metadata documents contains meta data of all elements in the service. You can see the metadata of a service by simple adding “$metadata” to the service URI.



Advantages of an ODATA

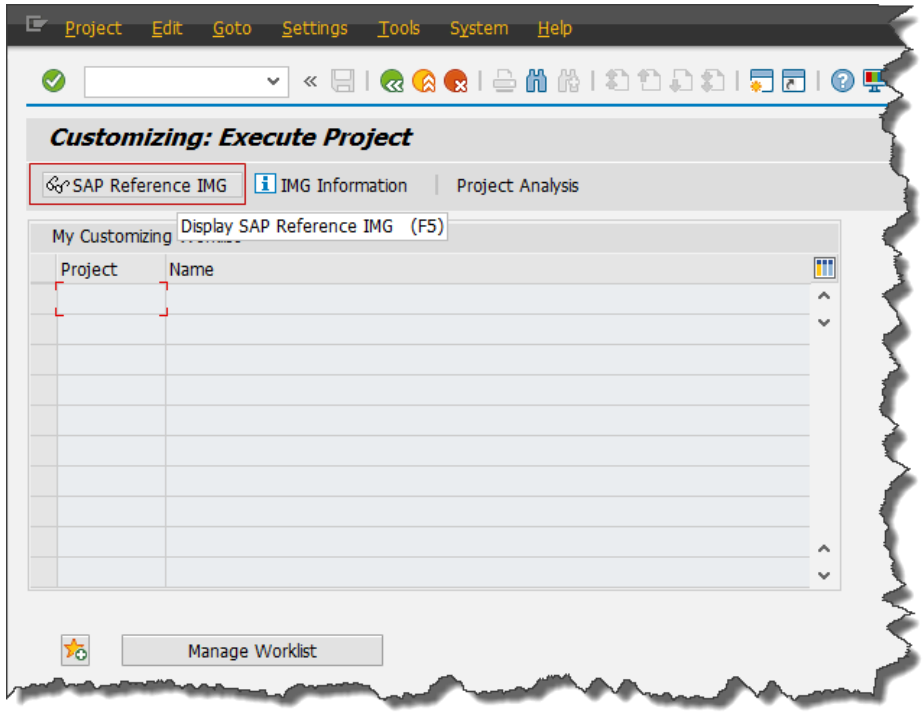
- Using SAP OData provides us following advantages:
- It helps to obtain human readable results i.e. you can use your browser to see the output data
- It is very easy and relatively fast to access data
- It uses all the standards of web protocols i.e. GET, PUT, POST, DELETE, and QUERY
- It uses Stateless Applications: It means Server does not save any data of Client (e.g. UI5 Application) and treats every OData call as a new call
- It receives data in form of related pieces of information, one leading to another: It is an interaction pattern known as “alert-analyse-act”, “view-inspect-act”, or “explore & act”. According to this pattern not all data are loaded together, and a user analyses a data and reaches its required information after navigation. In this way the data loads quickly and correctly.

Different elements in SAP OData service

- **Entity Type:** Entity is like work area which holds empty or one record data. As we have different fields in work area here also we have different fields and we call them as Properties.Each Entity should have at least one key field.
- **Entity Set:** Entity Set is a collection of same entity types. It is like internal table which holds n records of same type. For example list of sales orders is a Entity Set.
- **Property:** It represents a primitive data type element.It is like a single field in a work area or single column in a table. one or more properties are used to create an Entity Types.
- **Association:** It defines the relation between different entity types.For example if we have two entity types one for Sales Order header and other for Sales Order Item we can build the association between these two entity types with cardinality.
- **Navigation property:** Entity Types include one or more navigation properties.It is specific type which acts like a link to the other Entity types based on cardinality provided in the Association property. To create the navigation property for an entity type we need first define the association between those entity types.

How to Activate SAP Netweaver Gateway

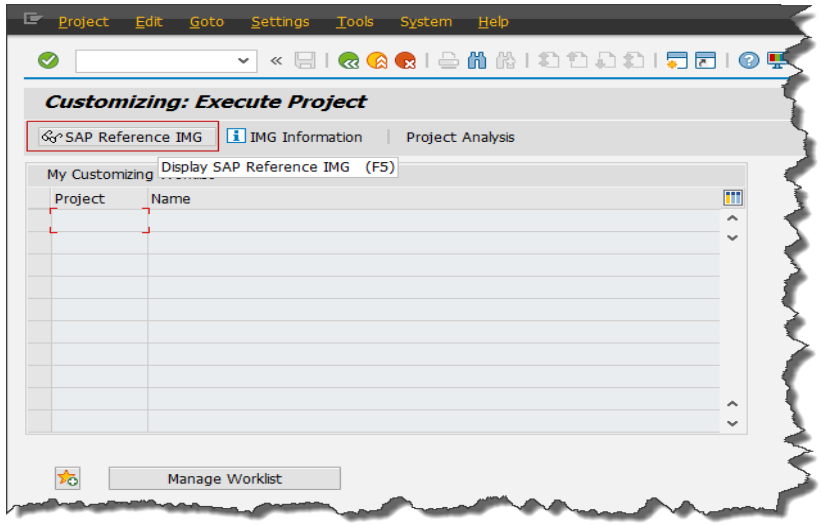
1. Go to SPRO >> SAP Reference IMG



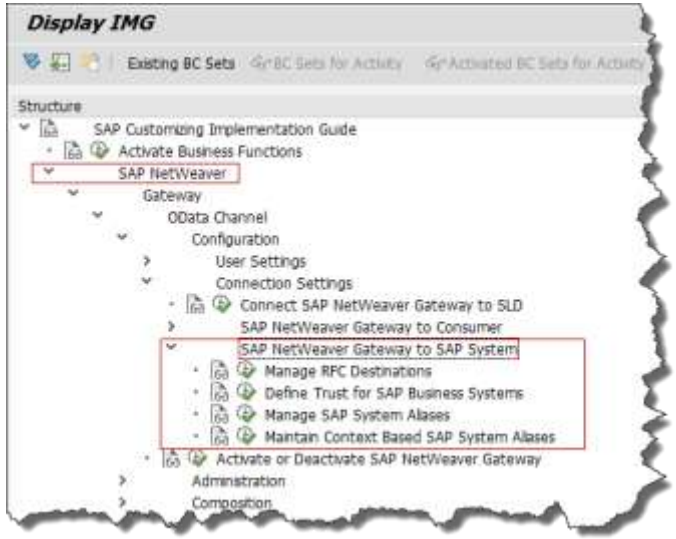
2. Expand *SAP Netweaver >> Gateway >> ODATA Channel >> Configuration >> Activate or Deactivate SAP Netweaver Gateway*
3. Click on Activate or Deactivate button to activate or deactivate SAP Netweaver Gateway.

How to Connect SAP Gateway to Backend Systems

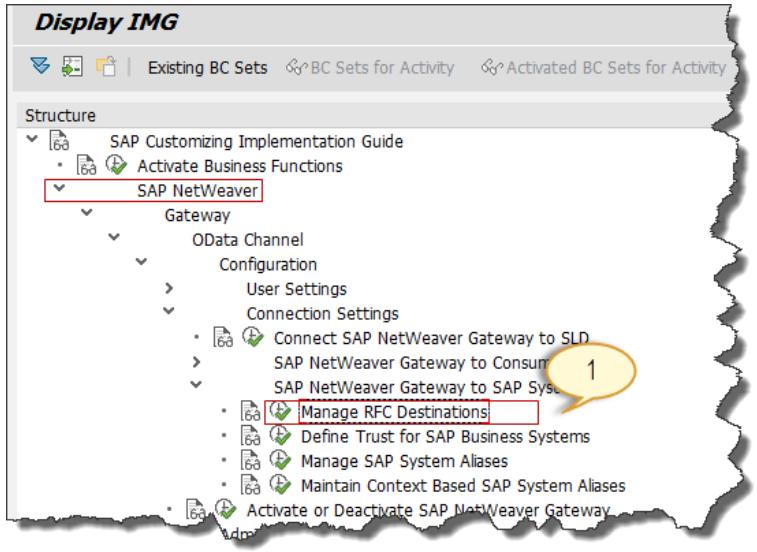
1. Go to SPRO >> SAP Reference IMG



2. Expand SAP Netweaver >> Gateway >> ODATA Channel >> Configuration >> Connection Settings >> SAP Netweaver Gateway to SAP System.



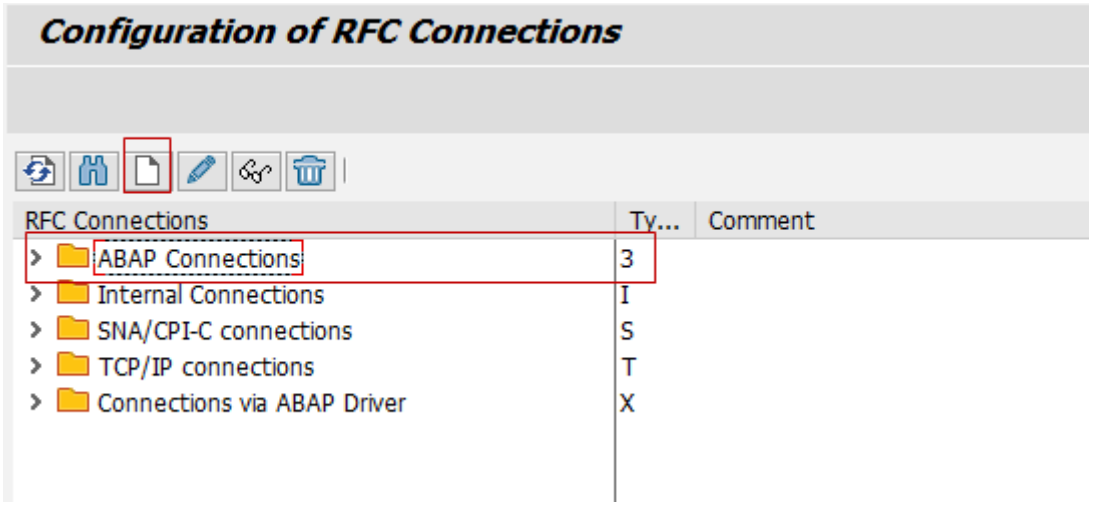
3. Execute “Mange RFC Destination”



4. Create a new ABAP RFC Connection to SAP Back-end Systems (ERP, SRM, CRM etc.).

   If you choose Hub Deployment, create a RFC connection to connect to SAP ERP system.

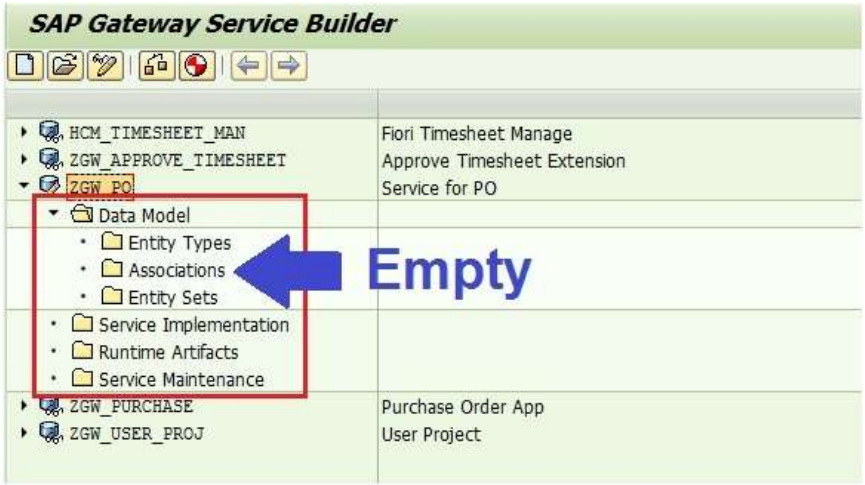
   If you choose Embedded Deployment, create a RFC connection to itself.



Simple Demo on ODATA Creation

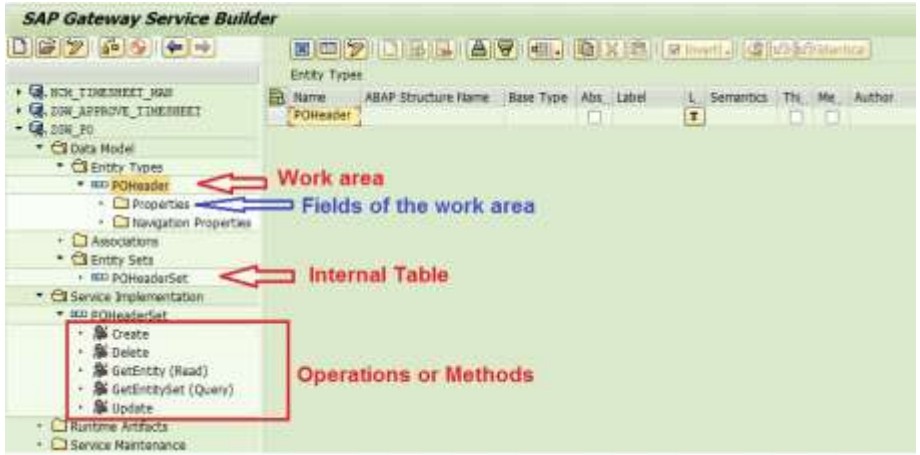
- Go to t-code **SEGW** (SAP Gateway Service Builder).Hit the Create Icon and provide the name of the Project, description and package (or local) and save it.

- The project gets created with four folders, namely ***Data Model, Service Implementation, Runtime Artifacts and Service Maintenance.*** Please take note that Data Model further has three sub-folders viz ***Entity Types, Associations and Entity Sets***. All the folders are empty by default.



- Entity Type is our very own structure (or a work area (holds just one row)). And you guessed it right, Entity Set is an internal table (holds more than one entity/rows).
- Let us create our first structure

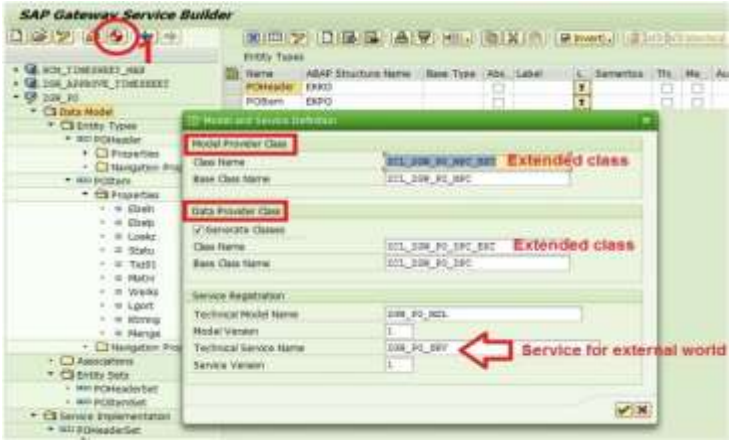
- Right click on Entity Types folder and select “Create”, provide the name you like and do not forget to tick the checkbox “Create Related Entity Set”. For our example, POHeader is the structure(work area) while POHeaderSet is our internal table.



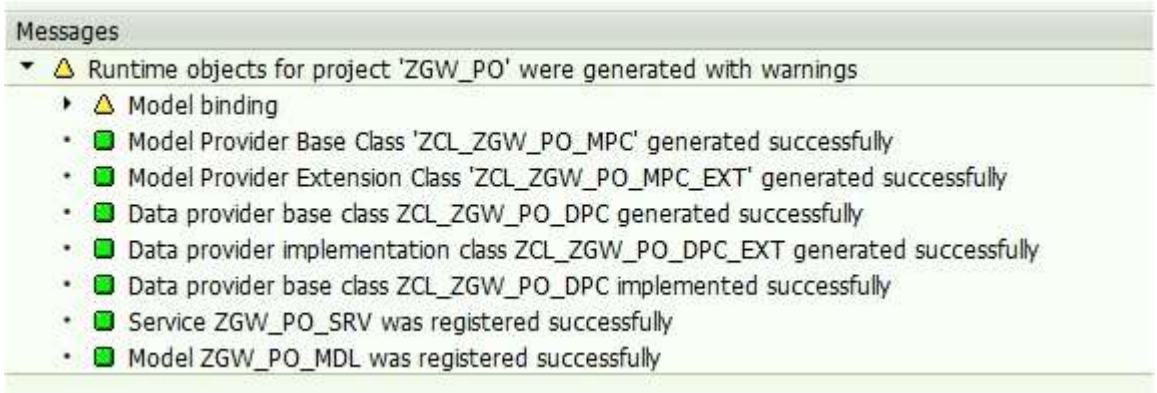
Check the Service Implementation folder has POHeaderSet Operations auto generated. *These are ABAP Methods which would be triggered when the relevant endpoints would be called*.

Implement/Register the Service

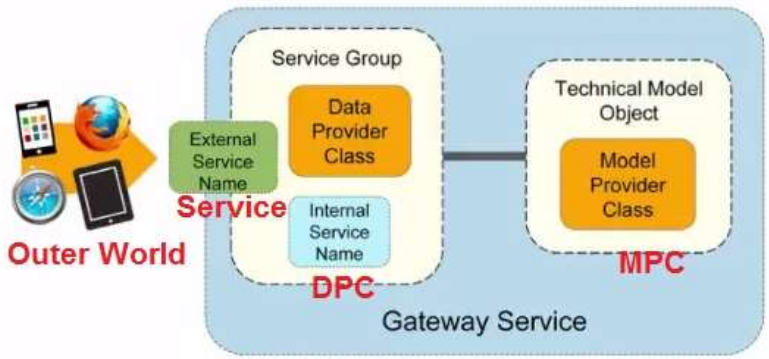
Let us go ahead and generate and register our service



- Hit the ***Generate*** icon and hit ok. Provide the package and transport number or save it as local. You should get the success message as shown below.

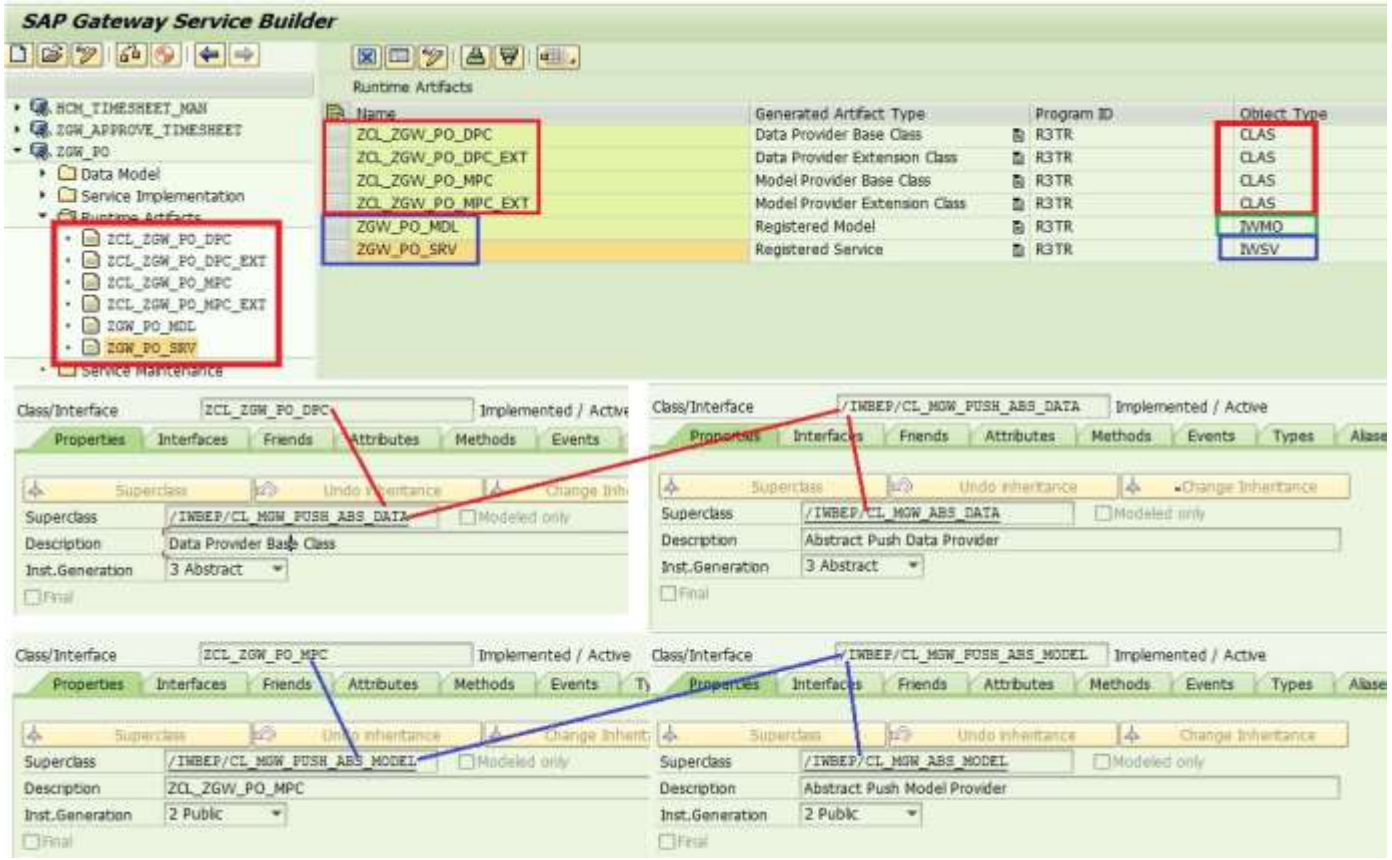


- Please note the ***Technical Service Name is the actual Service which the external system needs to call***. Two classes, ***Model Provider Class (MPC)*** and ***Data Provider Class (DPC)*** are also generated along with Base and Extended Class.



- **DPC and MPC are connected by Configuration (Not Coding)**

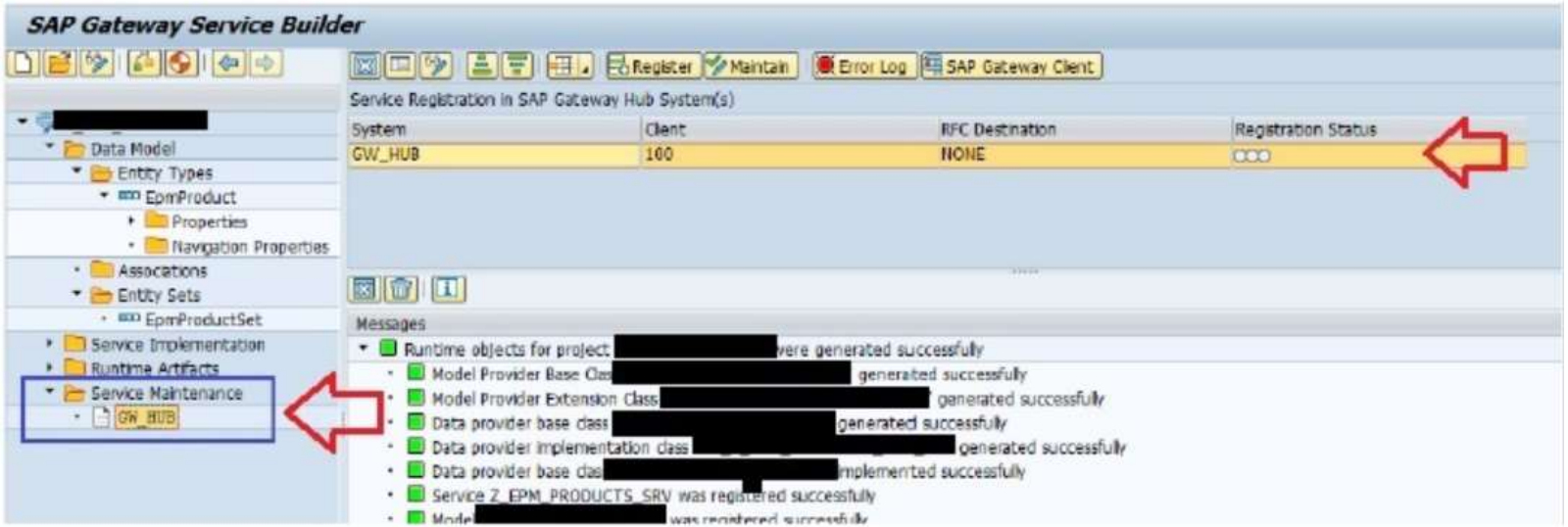
- Model Provider Class inherits from /IWBEP/CL_MGW_ABS_MODEL and Data Provider Class inherits from /IWBEP/CL_MGW_ABS_DATA. The below image shows the relationship between the generated classes and their superclasses (parents).



- Model Provider Class - This is used to define model. you can use the method Define to create entity, properties etc using code based implementation. you rarely use MPC extension class.
- Data Provider Class - used to code your CRUDQ methods as well as function import methods. you write all your logic in redefined methods of DPC extension class.

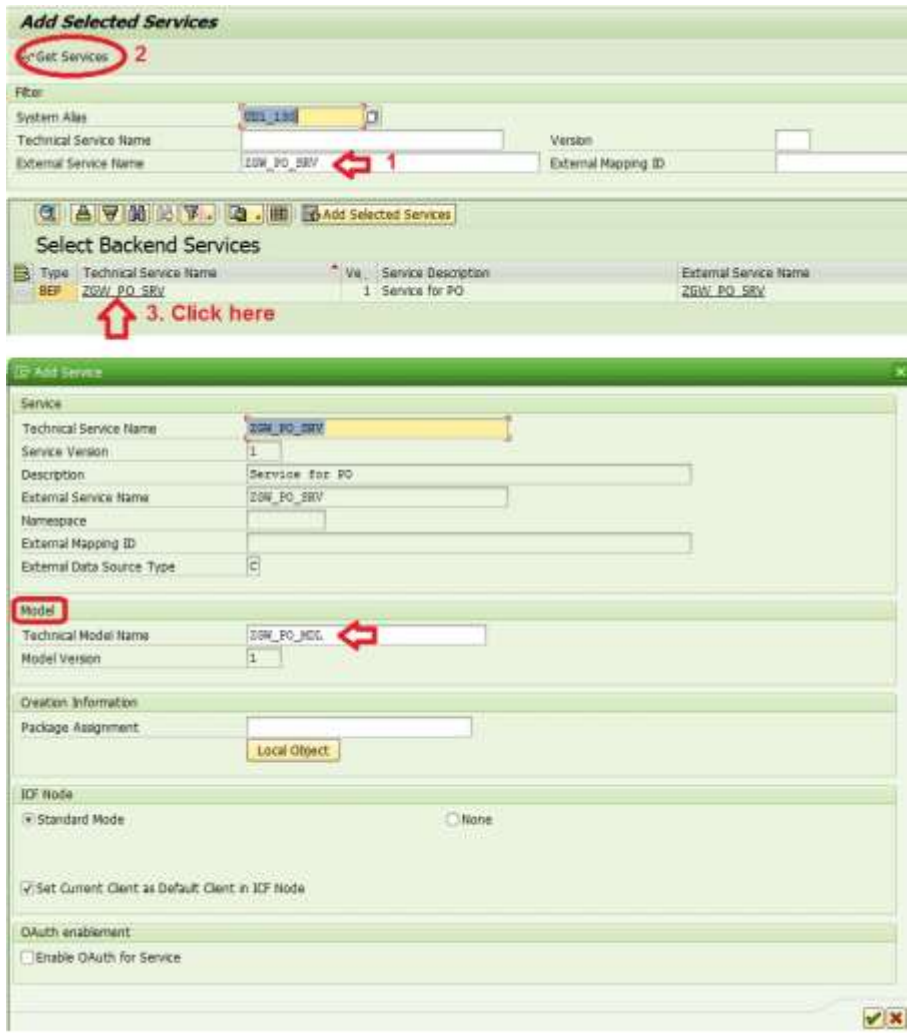
Add Service to Service Catalog (Register the Service to Gateway Hub)

- We have implemented the Service, now we need to add the service to the ***Service Catalog.***



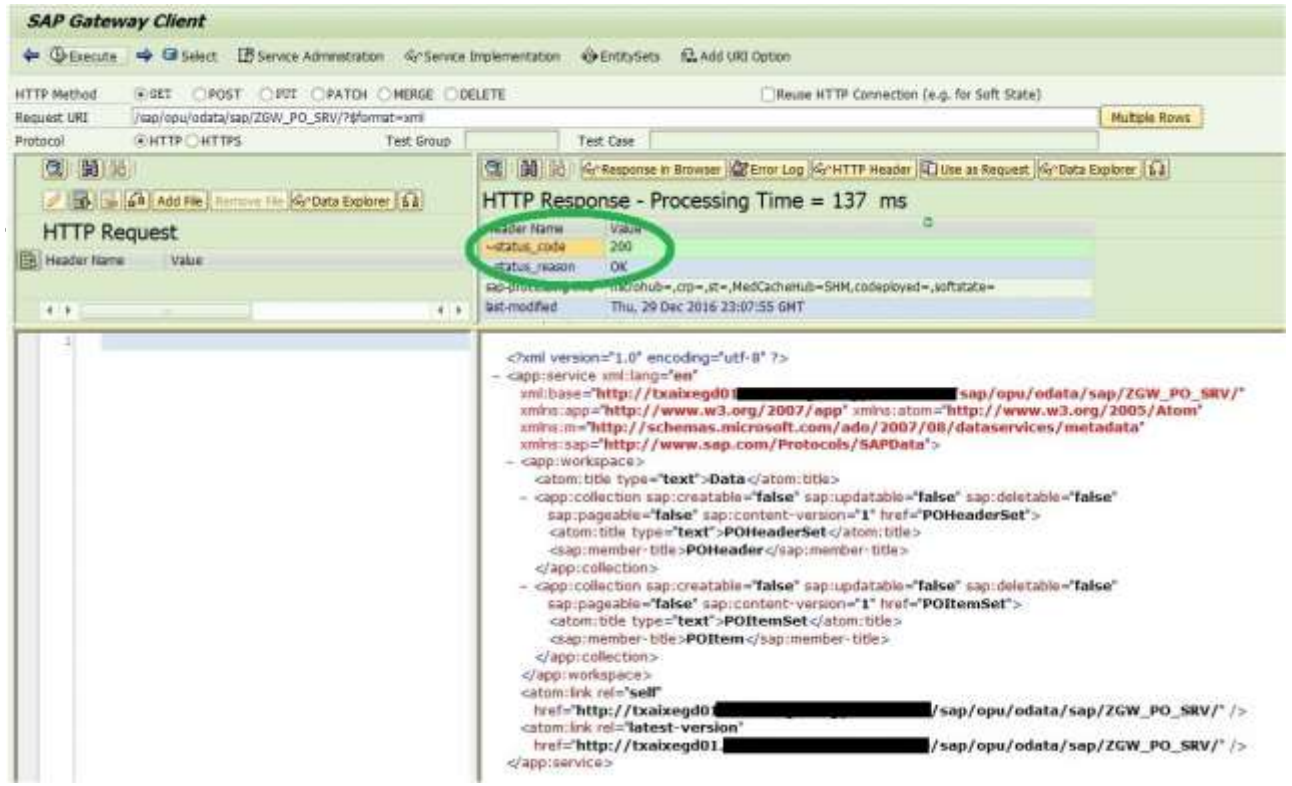
- Go to your ***Gateway Hub (Front-end system)*** and execute t-code /n/IWFND/MAINT_SERVICE

- Add the service to the Service Catalog, only then is our service available for the outer world to access**.**



- Hit the Add Service button, provide your backend system alias and external service name (our case ZGW_PO_SRV). You will get the service you created in the backend. Click on it and it would show the Service (technical/external) name along with the Technical name of the Model (ZGW_PO_MDL) and hit save. Go back to the Service Catalog screen.

- Let us test it using **SAP Gateway Client**. Or use t-code /IWFND/GW_CLIENT (remember this t-code as well). You can also test by Call Browser option. For now, we will use SAP Gateway Client option. status code is 200 i.e success.



- Message Status 2* is Success message, 5* is Error message