ERP-MDM Integration through PI – Part 1: IDoc to XML Conversion

Applies to:

SAP NetWeaver 2004s/ MDM 5.5 SP 06. For more information, visit the Master Data Management homepage.

Summary

This article deals with the step by step procedure to be carried out in PI in order to consolidate master data from ERP system to MDM.

Author: Shweta Singh

Company: Satyam Computer Services Ltd.

Created on: 06 November 2008

Author Bio

Shweta Singh has been associated with Satyam for 25 months and has been a part of MDM practice since March 2008.She is certified in XI and has completed her Bachelor's degree in Information Technology.

Table of Contents

Introduction

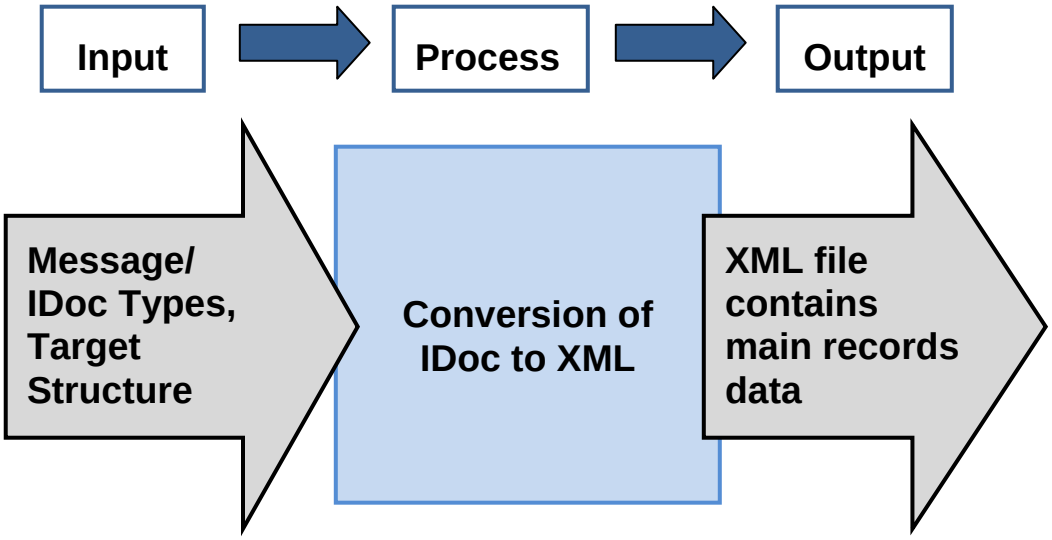
This article provides a step by step guide on the various configurations that need to be done in the PI system to send IDocs from the ERP system to MDM. This is the first part of the article in which I have tried to explain IDoc to XML conversion in PI for consolidating master data. The second part will have the details about the XML to IDoc conversion in order to harmonize the data back to the ERP system from MDM.

Business Scenario

The scenario consists of 2 systems, ERP and MDM (Master Data Management) that are required to exchange messages through PI. The container that contains the message is an IDoc (Intermediate Document). The IDoc consisting of the Material, Customer or Vendor records is extracted from the ERP system, converted into the XML format and posted into the desired folder in the MDM system.

Pre requisite: Make sure that before starting with the PI specific configurations, the Integration(ALE) Settings between the ERP and the MDM system to establish connection between the 2 systems and the configuration of Software Landscape Directory(SLD) in PI are already done.

Conversion of IDOC to XML



After the initial settings (ALE and SLD configurations) for the connection between the ERP and the MDM systems have been done, next we need to design and configure the scenario in the PI system.

For this we require to configure the two major components of the Integration Builder namely the Integration Repository (Design) and the Integration Directory (Configuration).

Integration Repository:

Definition/ Purpose: It is a component of Integration Builder wherein we define the objects like the source and the target structures, the interfaces, message mappings and interface mappings.

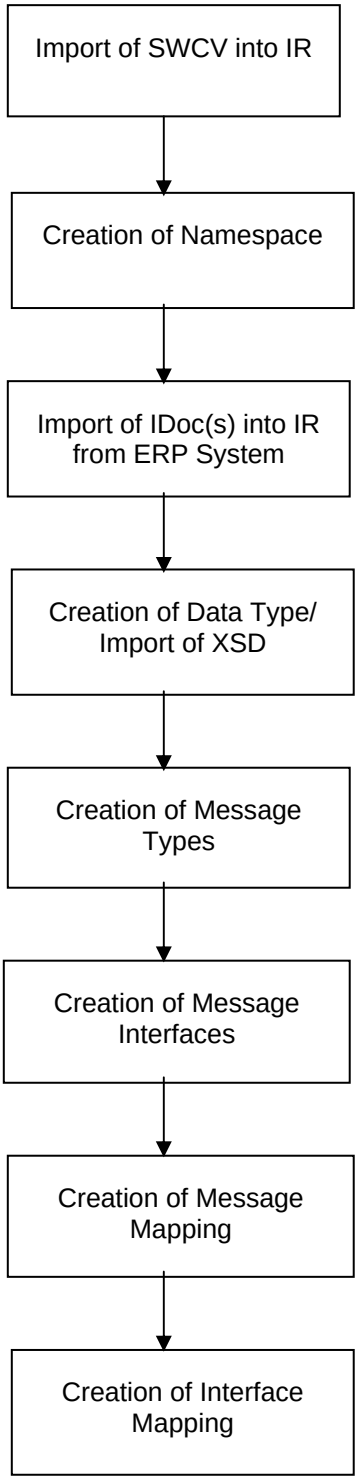
Prerequisite: The objects such as Software Product and Component, Technical and Business Systems are configured in Software Landscape Directory.

Integration Directory:

Definition/ Purpose: It is a component of Integration Builder where we configure the scenario designed in the Integration Repository. Here, we import (or create) objects like the sender and receiver systems from the SLD, configure adapters etc.

Prerequisite: The design objects relevant to the process are defined in the Integration Repository.

Integration Repository



Steps:

1. Import of SWCV from SLD into IR
2. Creation of Namespace
3. Import of IDoc from ERP System
4. Creation of Data Types
5. Creation of Message Types
6. Creation of Message Interfaces
7. Creation of Message Mapping
8. Creation of Interface Mapping

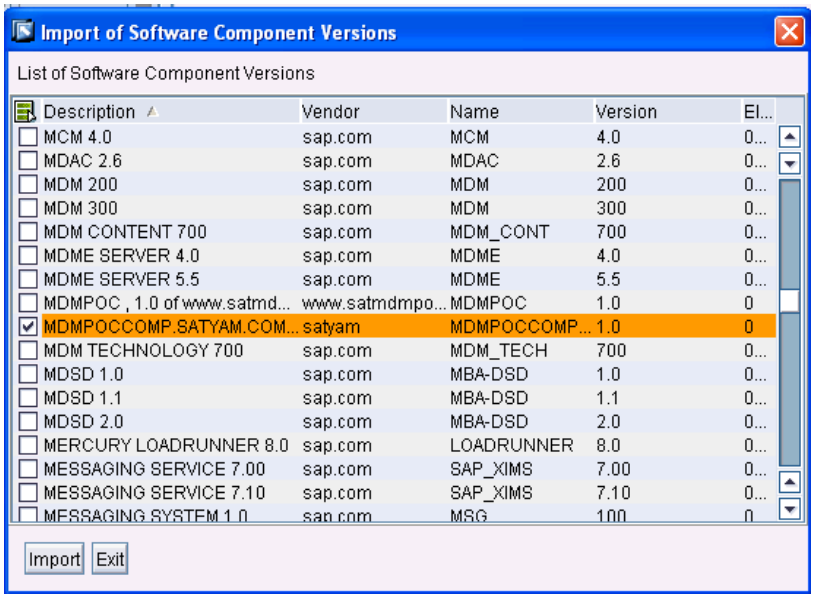
1. Import of objects and Creation of namespace in IR

1.1. Import of Software Component Version(SWCV) from SLD

1.1.1. Open the Integration Repository (Design) page by giving the appropriate User ID and Password.

1.1.2. Click on *Tools->Transfer from System Landscape Directory->Import Software Component Versions*.

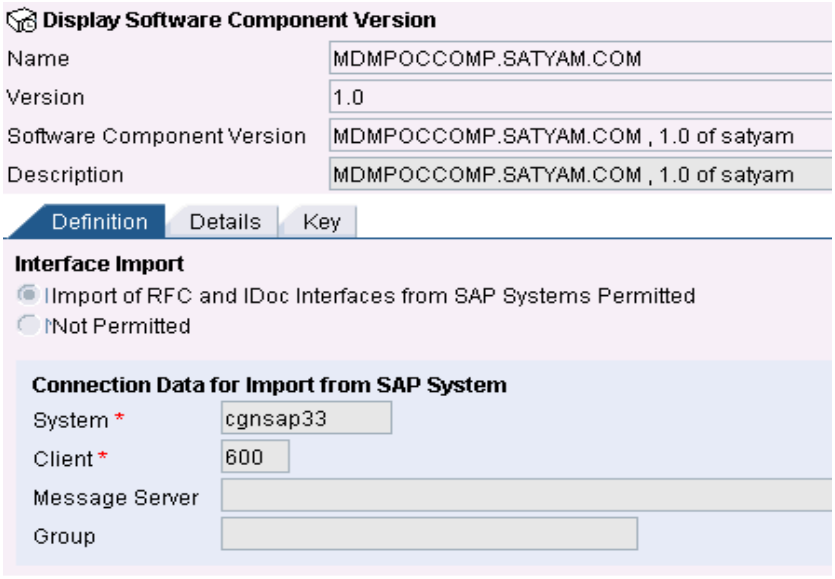
1.1.3. From the drop down list of the Software Component Versions created in the SLD, click on your created SLD and it will be imported into IR. You can now start creating the design objects in the SWCV imported.

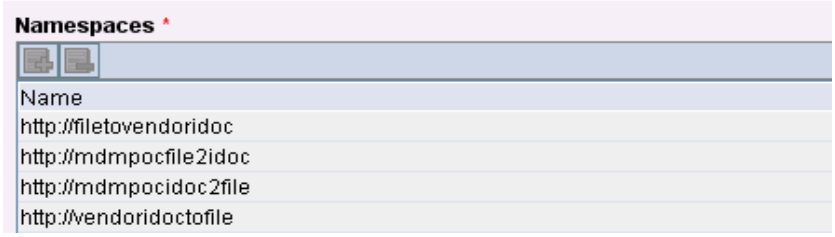


1.2. Creation of namespaces in the imported SWCV.

1.2.1. After importing the SWCV into IR, next step is to create a namespace that will hold the design objects together.

1.2.2. For this, open the SWCV, under the *Namespaces* tab, create a namespace and give it a suitable name. There is no standard naming convention as such for the namespace but usually it starts with *http:// or urn:* followed by the name.



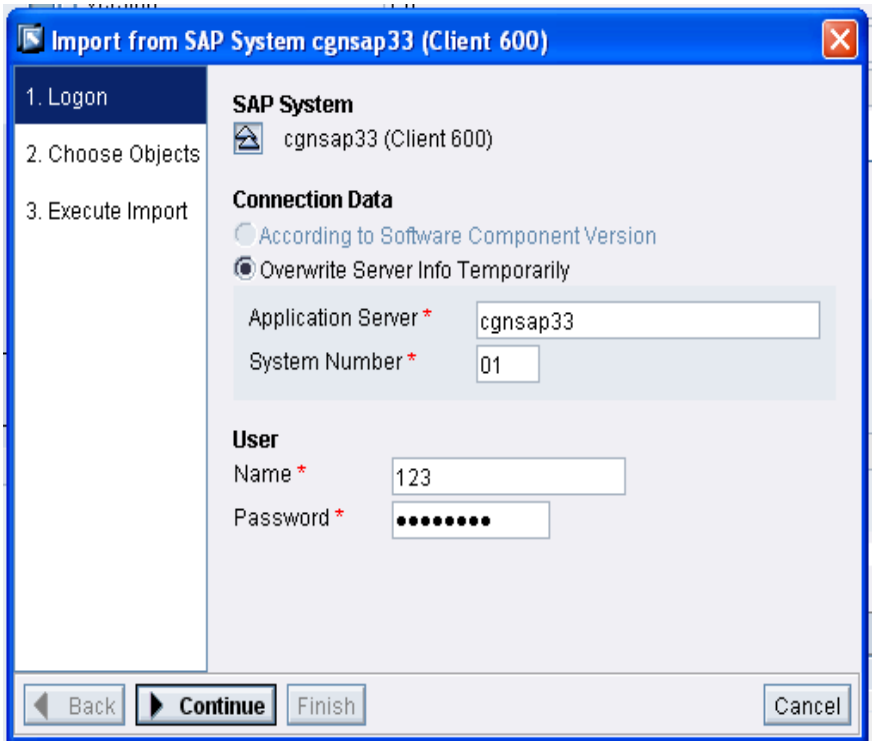


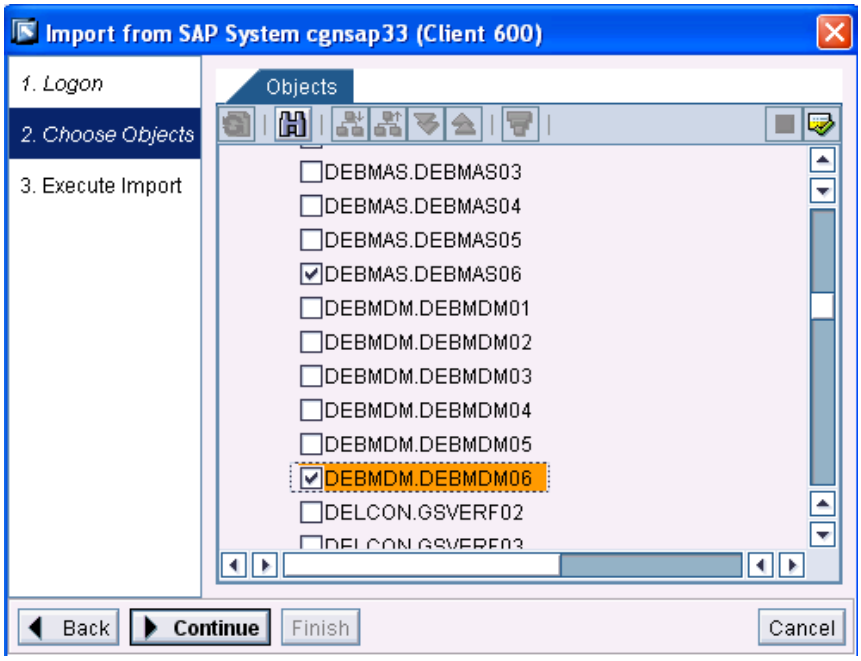
1.3. Import of IDOCs from the ERP system according to the requirement

1.3.1. Now, the IDoc that will act as the source structure has to be imported in the SWCV from the ERP system. For this too, the appropriate Login credentials are required. (In case you don't have the rights to import an IDoc from the ERP system, you can ask your BASIS team to do it.)

1.3.2. Under the SWCV, click *on Imported Objects->Import of SAP Objects*. Give the *Application Server* name , *System Number* of the ERP system you are importing the IDoc from and your User ID and Password in the same system and *Continue.*

1.3.3. From the drop down list, select the IDoc you want to import and click on *Finish*. You will have the IDoc imported under *Imported Objects->IDocs.*


Import from SAP System cgnsap33 (Client 600)
1. Logon
2. Choose Objects
3. Execute Import
SAP System
cgnsap33 (Client 600)
Connection Data
According to Software Component Version
Overwrite Server Info Temporarily
Application Server *
cgnsap33
System Number *
01
User
Name *
123
Password *
Back
Continue
Finish
Cancel


Import from SAP System cgnsap33 (Client 600)
1. Logon
2. Choose Objects
3. Execute Import
Objects
DEBMAS.DEBMAS03
DEBMAS.DEBMAS04
DEBMAS.DEBMAS05
DEBMAS.DEBMAS06
DEBMDM.DEBMDM01
DEBMDM.DEBMDM02
DEBMDM.DEBMDM03
DEBMDM.DEBMDM04
DEBMDM.DEBMDM05
DEBMDM.DEBMDM06
DELCON.GSVERF02
Back
Continue
Finish
Cancel

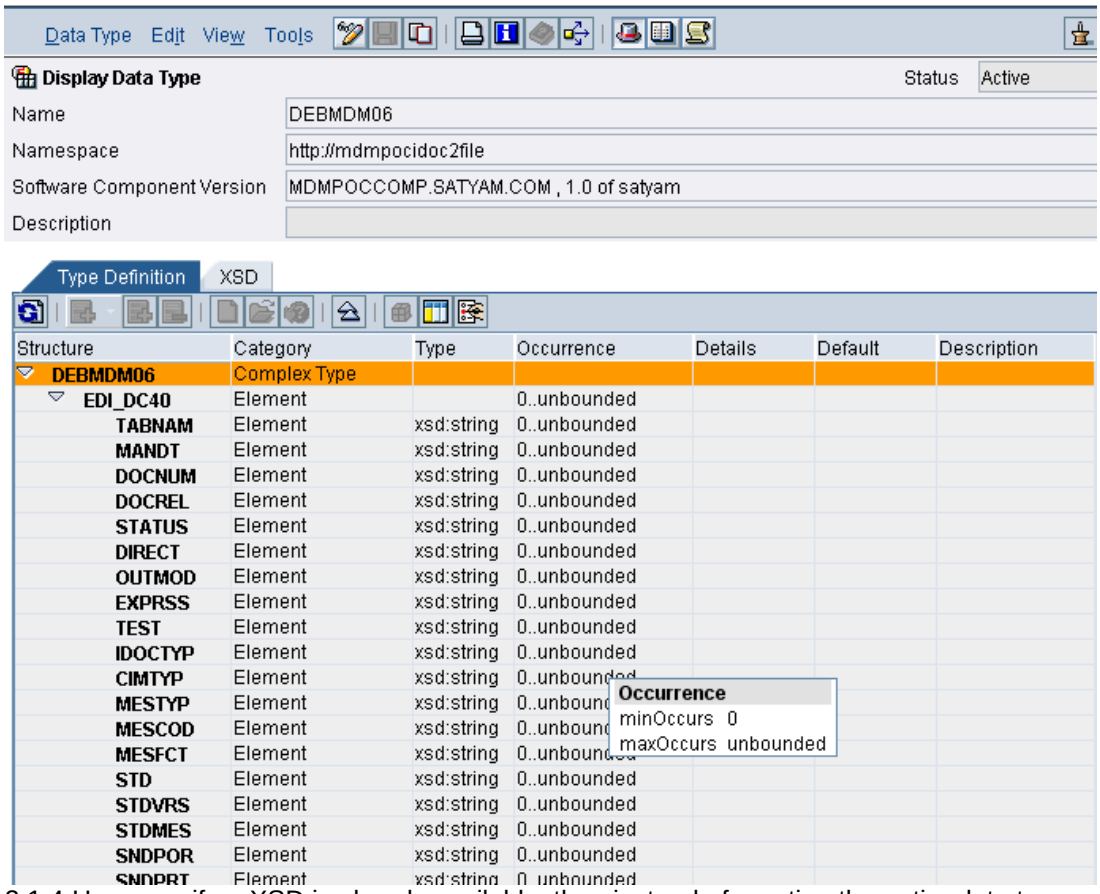
2. Designing Interface Objects

2.1. Creation of Data Types/ Import of Target XSD

2.1.1. Here, only one data type has to be created and that is for the target structure. The source structure will be the IDoc itself hence no need to create any structure for it.

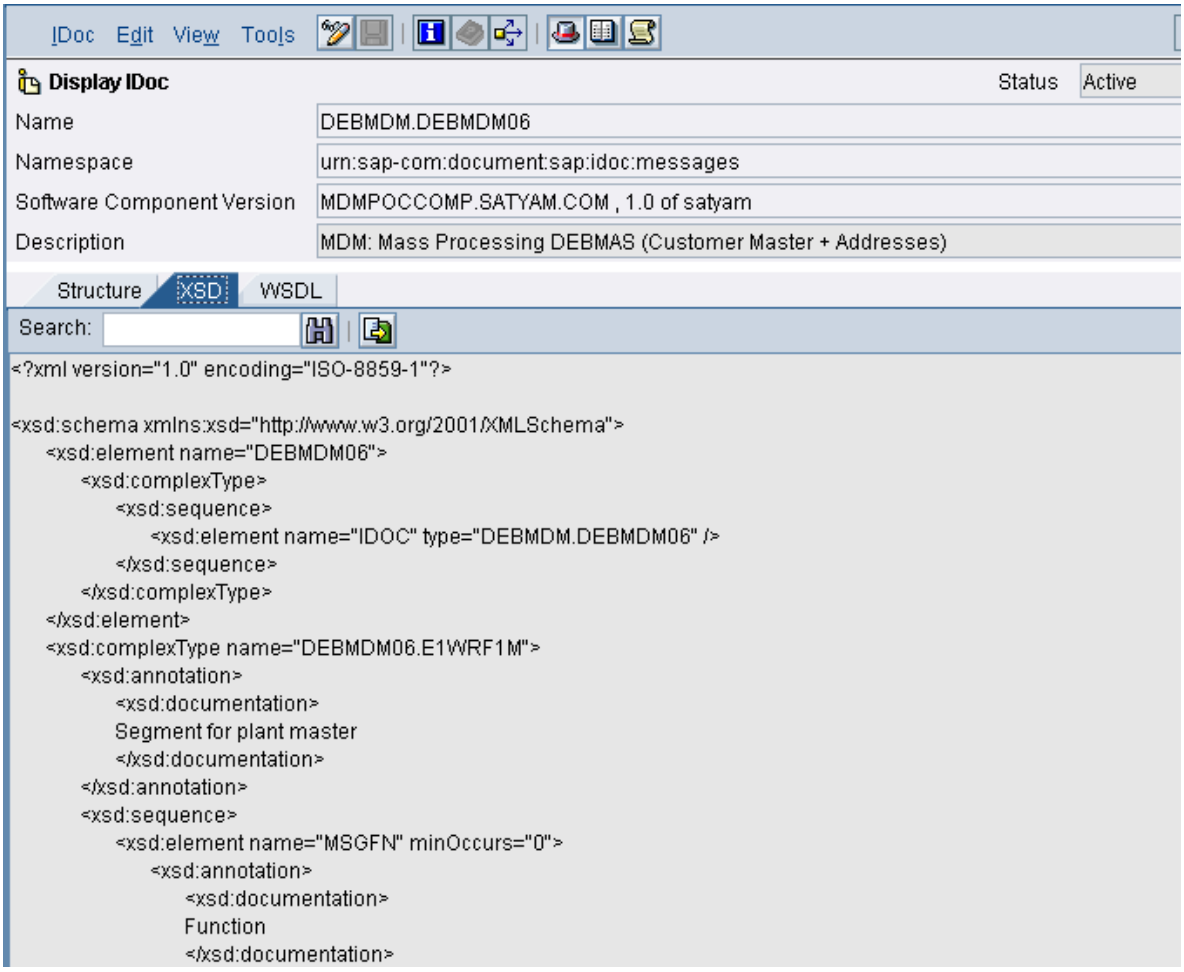
2.1.2. In order to create the Data Type, go to *Interface Objects->Data Types->New*.

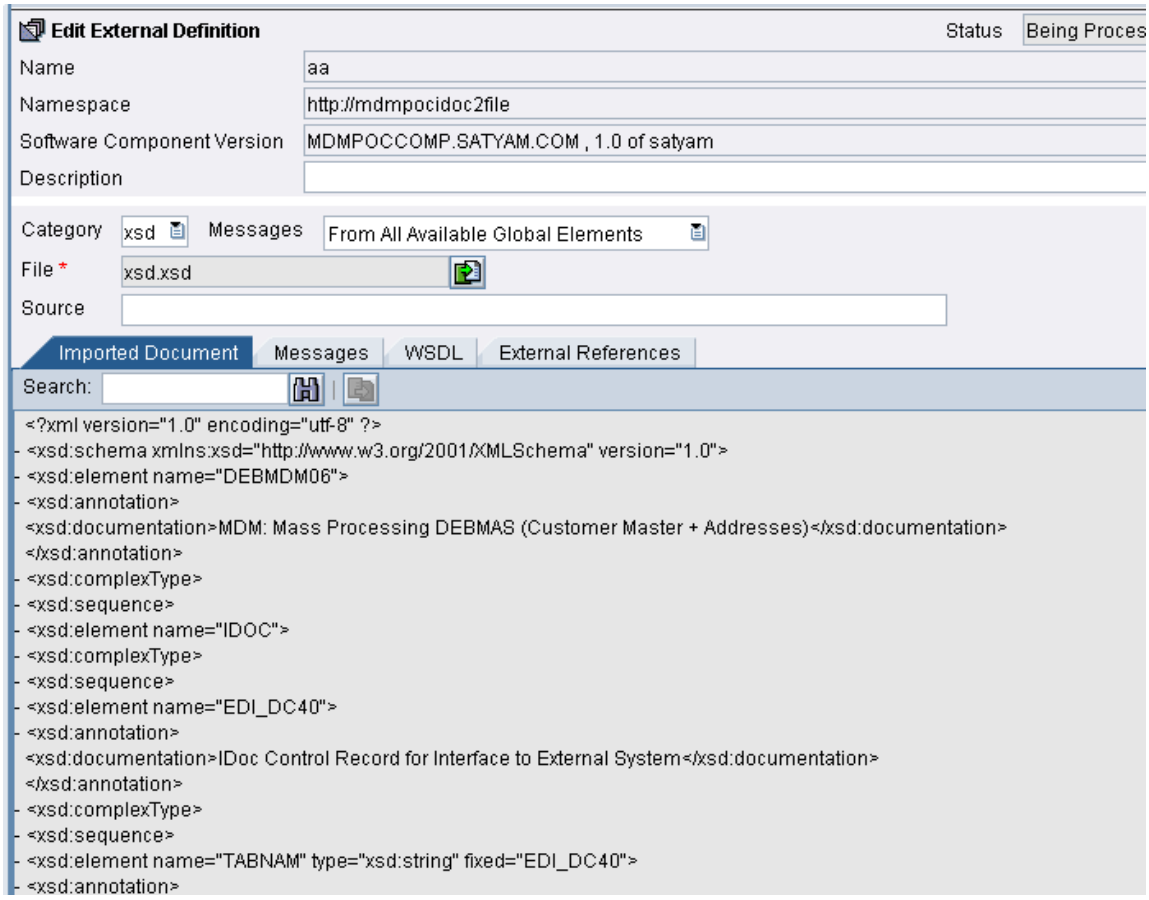
2.1.3. Give the Data Type a suitable name and create the structure with the appropriate segments and fields. Also take care of the minimum and maximum allowed occurrences of each field.



2.1.4. However, if an XSD is already available, then instead of creating the entire data type manually, the XSD just needs to be imported into the PI system under *External Definitions* which will serve as the target structure.

2.1.5. Also, if the source (IDoc) and the target (XML) structures are exactly the same (i.e. both have exactly the same segments and fields), the XSD can also be created from the imported IDoc. For this, open the IDoc, click on the tab *XSD* and export the XSD to file(i.e. somewhere on the local machine)by clicking on the green button. Open your created namespace, *Interface Objects->External Definitions.* Click on *New,* give a suitable name to the External Definition (eg. MATMAS_ED, DEBMDM_ED), select the *Category* as XSD and the file you had exported on your machine from the *File* tab and save. This External Definition can now be used as the target data structure.


IDoc Edit View Tools
Display IDoc
Status Active
Name DEBMDM.DEBMDM06
Namespace urn:sap-com:document:sap:idoc:messages
Software Component Version MDMPOCCOMP.SATYAM.COM , 1.0 of satyam
Description MDM: Mass Processing DEBMAS (Customer Master + Addresses)
Structure XSD WSDL
Search:
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:element name="DEBMDM06">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="IDOC" type="DEBMDM.DEBMDM06" />
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<xsd:complexType name="DEBMDM06.E1WRF1M">
<xsd:annotation>
<xsd:documentation>
Segment for plant master
</xsd:documentation>
</xsd:annotation>
<xsd:sequence>
<xsd:element name="MSGFN" minOccurs="0">
<xsd:annotation>
<xsd:documentation>
Function
</xsd:documentation>

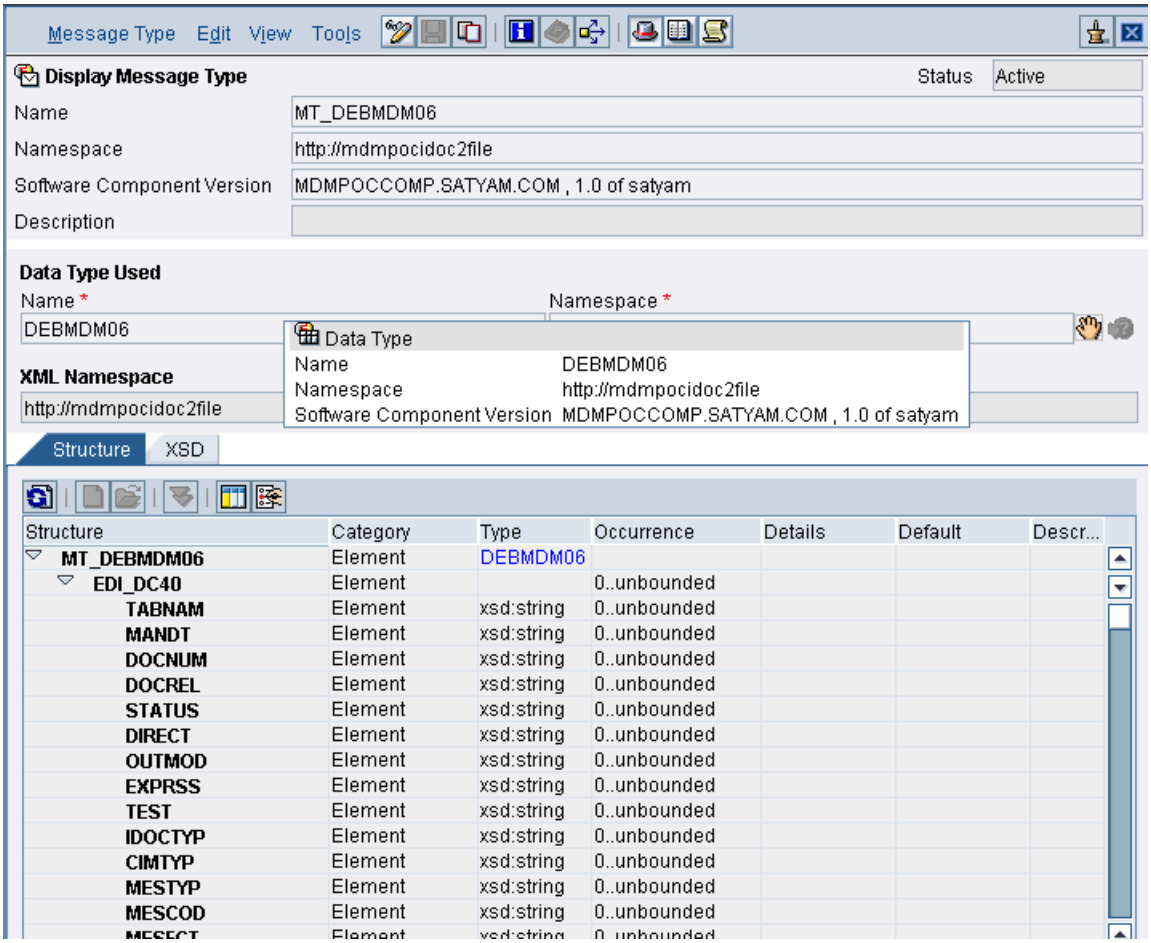


2.2. Creation of Message Types

2.2.1. A Message Type is created for the above Data Type.

2.2.2. Go to *Interface Objects->Message Types->New*, select the corresponding Data Type for which you are creating the Message type. Give the Message Type a suitable name and save.

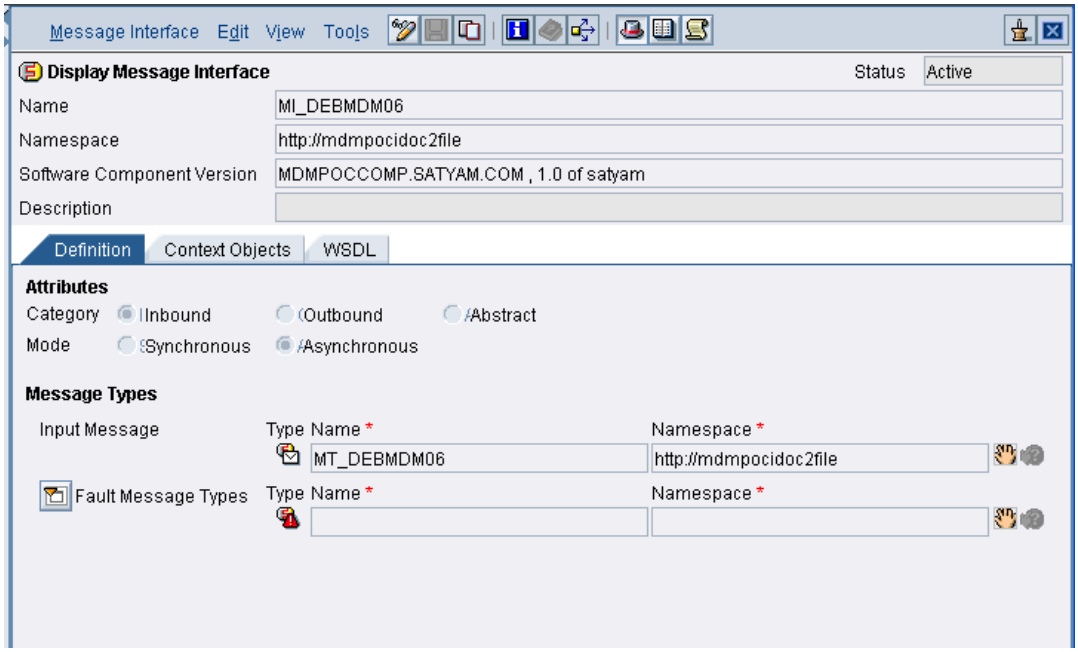
Note: A Message type doesn't need to be created for an IDoc because it itself acts as one. Also, if for the target structure, an External Definition is being used, in that case too, the message type doesn't need to be created.



2.3. Creation of Message Interfaces

2.3.1. A Message Interface is created using the above Message Type.
2.3.2. It has to be created only for the target structure if Data Type and the Message Type have been created for the target.
2.3.3. Go to *Interface Objects->Message Interfaces->New,* give the Interface a name and select the corresponding Message Type. Also, specify the direction of the interface as Inbound or outbound.
2.3.4. For the Message Interface that is created here, the direction will be Inbound since it will be receiving the message from XI.

***Note*:** For an IDoc , the Data Types , Message Types and Message Interfaces don't need to be created because IDoc itself acts as all of these.
Hence, in this scenario, we have only one Data Type, one Message Type and one Message Interface and they are only for the target message.



3. Designing Mapping Objects

3.1. Creation of Message Mapping

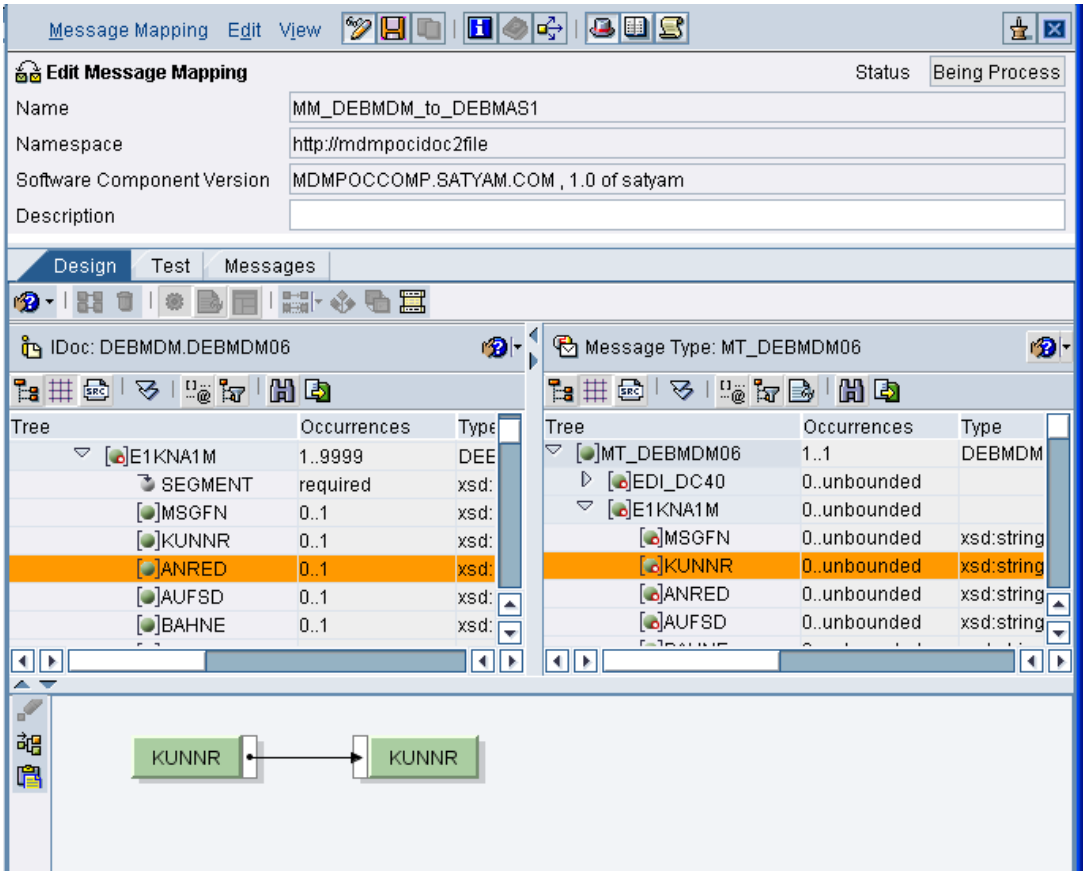
3.1.1. A mapping is done between the source (IDoc) and the target(Message Type or External Definition) structures.

3.1.2. In order to create the mapping, go *to Mapping Objects-> Message Mappings->New.* Give it a suitable name, select the Source Message Type and the Target Message Type.

3.1.3. Now according to the requirement, map some or all of the source fields with the corresponding target fields. Please note that there might be some mandatory fields that need to be mapped irrespective of whether required or not, without which the scenario will throw some errors.

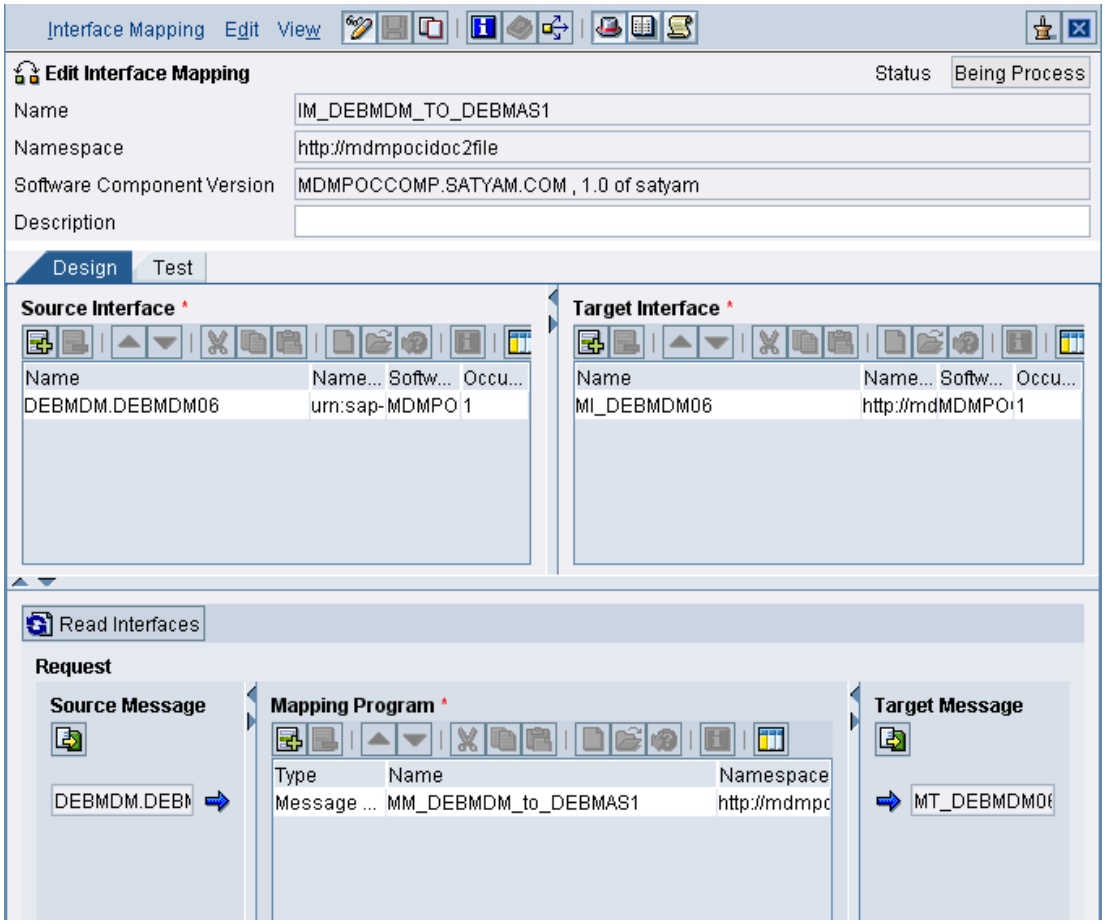
3.1.4. After finishing mapping of all the required fields, click on the *Test* tab to test the mapping. Give some arbitrary values for some of the fields and check if mapping is correct. If yes, you will get a message **Executed Successfully**.

3.1.5. Save the mapping.

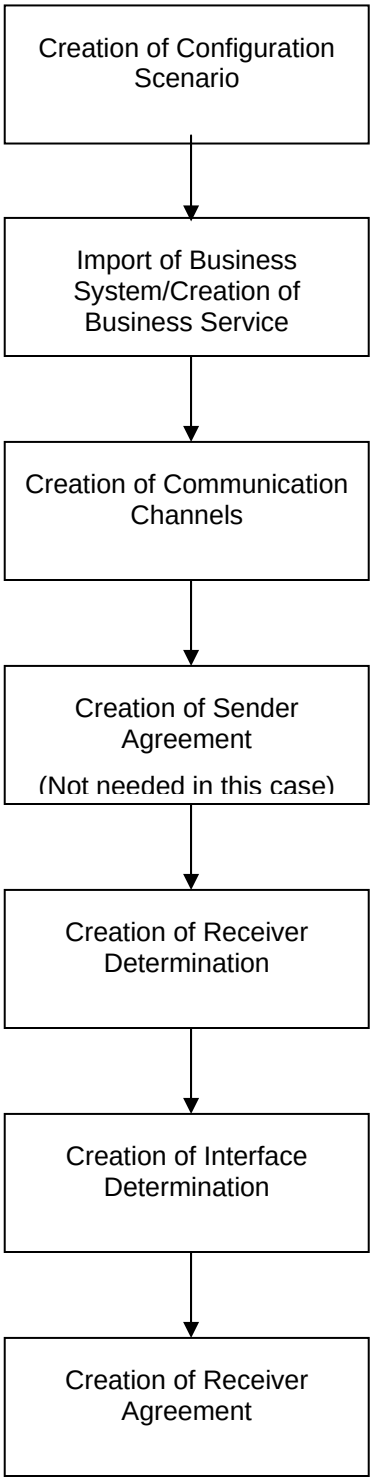


3.2. Creation Of Interface Mapping

3.2.1. After finishing the Message Mapping, next step is to create the Interface Mapping.
3.2.2. Go to *Mapping Objects->Interface Mapping->New.*
3.2.3. Give the Interface mapping a suitable name and choose the Source Interface and the Target Interface and click on *Read Interfaces.*
3.2.4. The source and the target messages along with the Mapping Program will be displayed. However, some times, the mapping Program doesn't get displayed on its own. In that case, you can choose it yourself from the drop down list.
3.2.5. Save the Interface Mapping.


Interface Mapping Edit View
Edit Interface Mapping
Status Being Process
Name IM_DEBMDM_TO_DEBMAS1
Namespace http://mdmpocidoc2file
Software Component Version MDMPOCCOMP.SATYAM.COM , 1.0 of satyam
Description
Design Test
Source Interface *
Name Name... Softw... Occu...
DEBMDM.DEBMDM06 urn:sap- MDMPO 1
Target Interface *
Name Name... Softw... Occu...
MI_DEBMDM06 http://md MDMPO 1
Read Interfaces
Request
Source Message
DEBMDM.DEBM
Mapping Program *
Type Name Namespace
Message ... MM_DEBMDM_to_DEBMAS1 http://mdmpc
Target Message
MT_DEBMDM0

Integration Directory

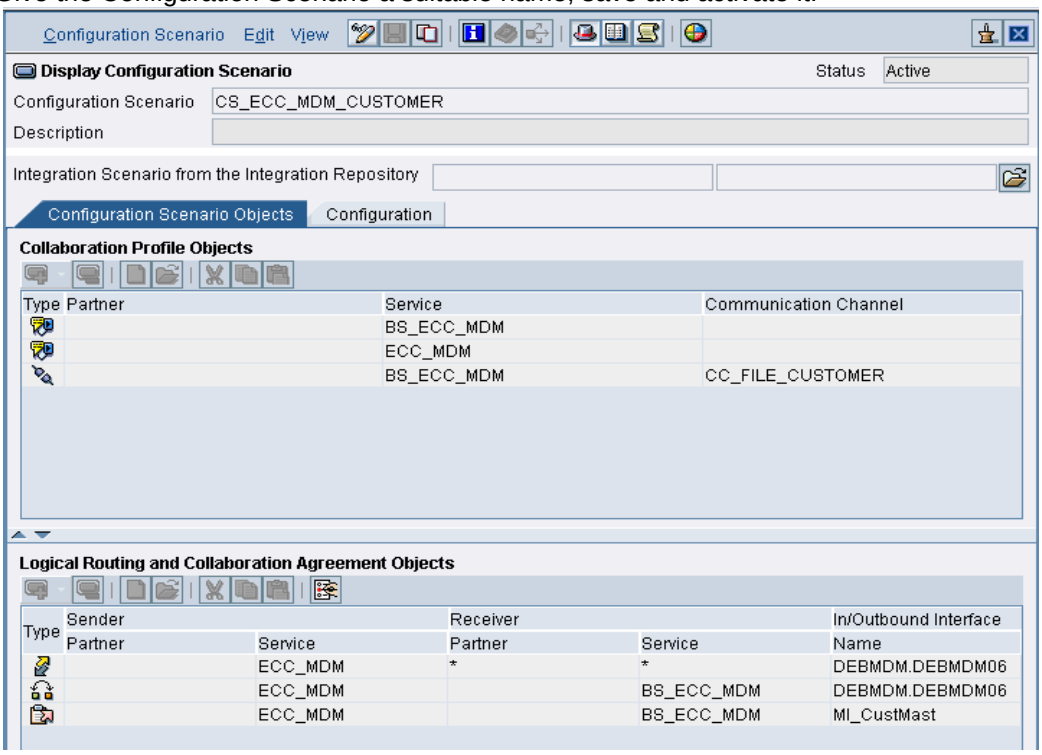


Steps:

1. Creation of Configuration Scenario
2. Import of Business System/Creation of Business Service
3. Creation of Communication Channels
4. Creation of Receiver Determination
5. Creation of Interface Determination
6. Creation of Receiver Agreement

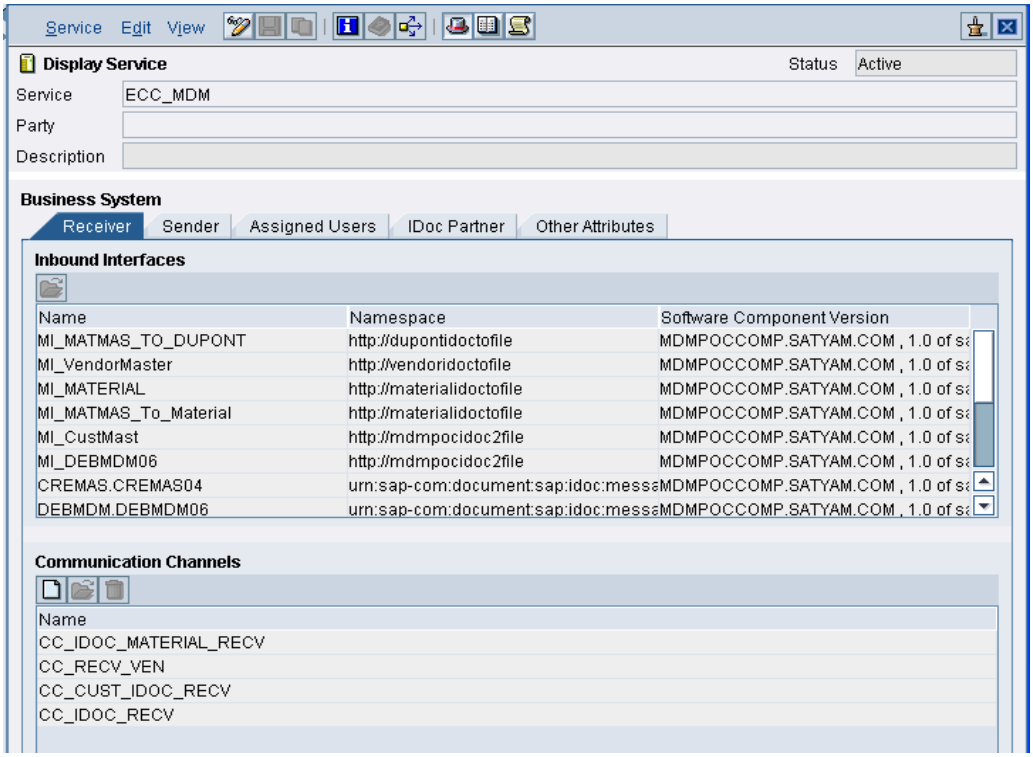
1. Creation of Configuration Scenario

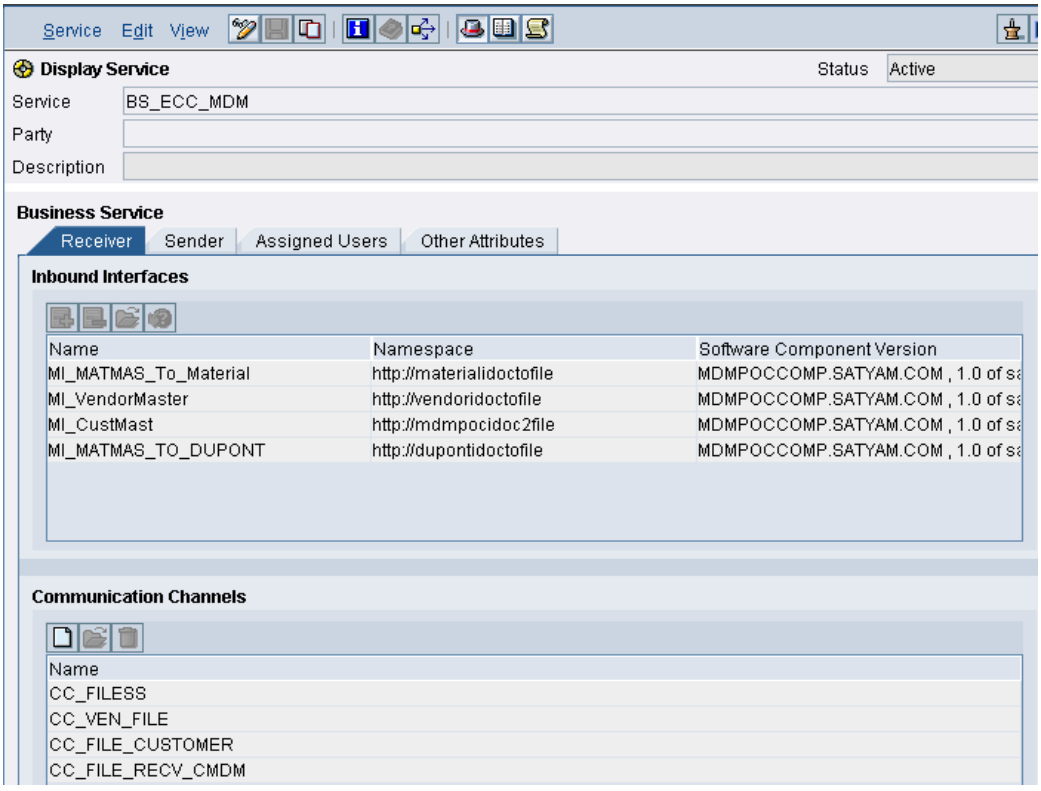
1.1. Open the Integration Directory (Configuration) page and under the *Scenarios* tab, click *New*.
1.2. Give the Configuration Scenario a suitable name, save and activate it.



2. Import of Business System / Creation of Business Service

2.1. After the scenario has been created, next step is to specify and configure the systems or the services participating in the scenario.
2.2. You can have 2 Business Systems or 2 Business Services for Sender and Receiver or one Business System for Sender and Business Service for Receiver or vice-versa.
2.3. If you have created a Business system already in the SLD, then you have to import it in the ID.
2.4. For this, click on *Service Without Party->Business System->Assign Business System.* Select your created Business System from the drop down list.
2.5. To create the Business Service, go to *Service Without Party->Business Service->New.*
2.6. Give the Business Service a name and select all the required parameters that need to be entered such as whether the Service will act as the sender or receiver etc.
2.7. Also, add all the inbound and outbound interfaces created in the Integration Repository earlier that will be required in the scenario (You don't need to add all the interfaces at once. You can add them as and when required in the scenario).


Service Edit View
Display Service
Status Active
Service ECC_MDM
Party
Description
Business System
Receiver Sender Assigned Users IDoc Partner Other Attributes
Inbound Interfaces
Name Namespace Software Component Version
MI_MATMAS_TO_DUPONT http://dupontidoctofile MDMPOCCOMP.SATYAM.COM , 1.0 of sa
MI_VendorMaster http://vendoridoctofile MDMPOCCOMP.SATYAM.COM , 1.0 of sa
MI_MATERIAL http://materialidoctofile MDMPOCCOMP.SATYAM.COM , 1.0 of sa
MI_MATMAS_To_Material http://materialidoctofile MDMPOCCOMP.SATYAM.COM , 1.0 of sa
MI_CustMast http://mdmpocidoc2file MDMPOCCOMP.SATYAM.COM , 1.0 of sa
MI_DEBMDM06 http://mdmpocidoc2file MDMPOCCOMP.SATYAM.COM , 1.0 of sa
CREMAS.CREMAS04 urn:sap-com:document:sap:idoc:messa MDMPOCCOMP.SATYAM.COM , 1.0 of sa
DEBMDM.DEBMDM06 urn:sap-com:document:sap:idoc:messa MDMPOCCOMP.SATYAM.COM , 1.0 of sa
Communication Channels
Name
CC_IDOC_MATERIAL_RECV
CC_RECV_VEN
CC_CUST_IDOC_RECV
CC_IDOC_RECV


Service Edit View
Display Service
Status Active
Service BS_ECC_MDM
Party
Description
Business Service
Receiver Sender Assigned Users Other Attributes
Inbound Interfaces
Name Namespace Software Component Version
MI_MATMAS_To_Material http://materialidoctofile MDMPOCCOMP.SATYAM.COM , 1.0 of sa
MI_VendorMaster http://vendoridoctofile MDMPOCCOMP.SATYAM.COM , 1.0 of sa
MI_CustMast http://mdmpocidoc2file MDMPOCCOMP.SATYAM.COM , 1.0 of sa
MI_MATMAS_TO_DUPONT http://dupontidoctofile MDMPOCCOMP.SATYAM.COM , 1.0 of sa
Communication Channels
Name
CC_FILESS
CC_VEN_FILE
CC_FILE_CUSTOMER
CC_FILE_RECV_CMDM

3. Creation of Communication Channels

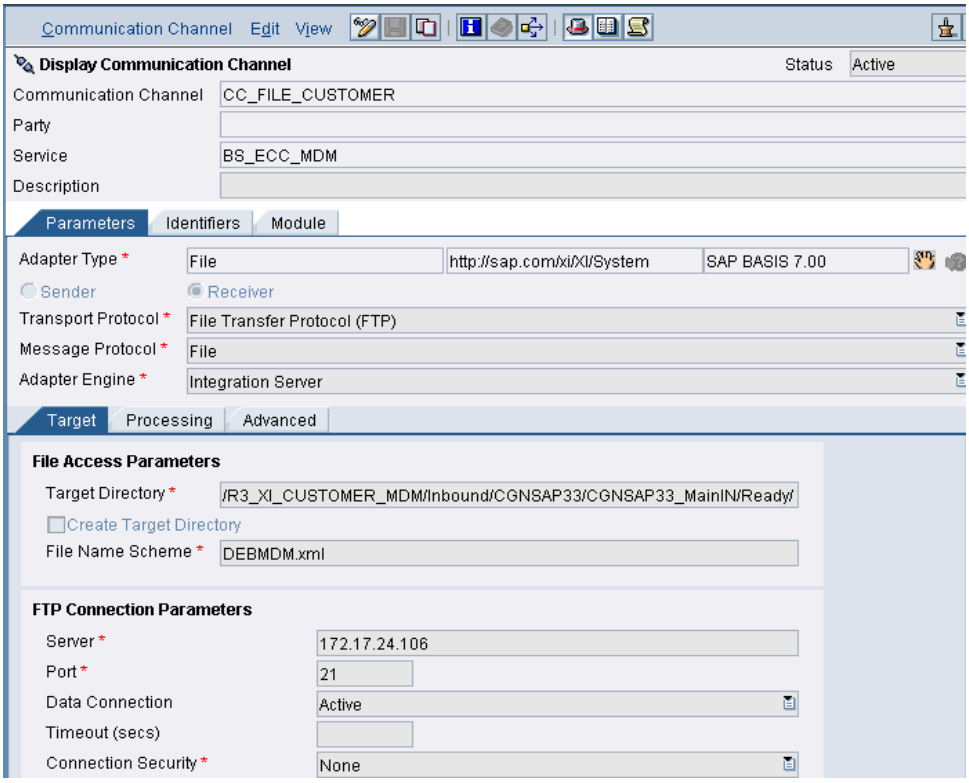
3.1. Here, two communication channels need to be created, one that will act as the Sender Communication Channel and the other as the Receiver Communication Channel.

3.2. The Sender Communication Channel will be created under the Sender System (or Service) and the Receiver Communication Channel will be created under the Receiver System (or Service).

3.3. Here in this case however, Sender Communication Channel is not required as the IDoc adapter is present in the ABAP stack.

3.4. For creation of the Receiver Communication Channel, go to *Business Service->Business Service Name->Communication Channel->New.*

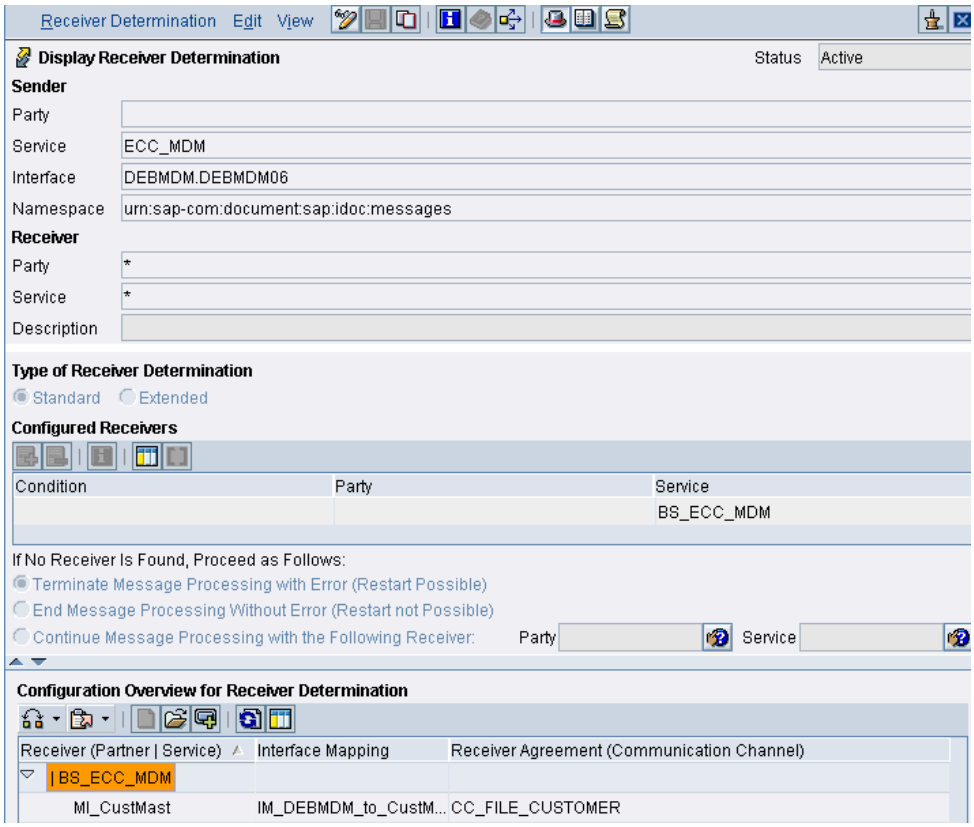
3.5. Give the channel a name, specify that it is the receiver communication channel and give the required parameters like the target path where the file in the XML format has to be placed, the *Transport Protocol* etc.



4. Creation of Receiver Determination

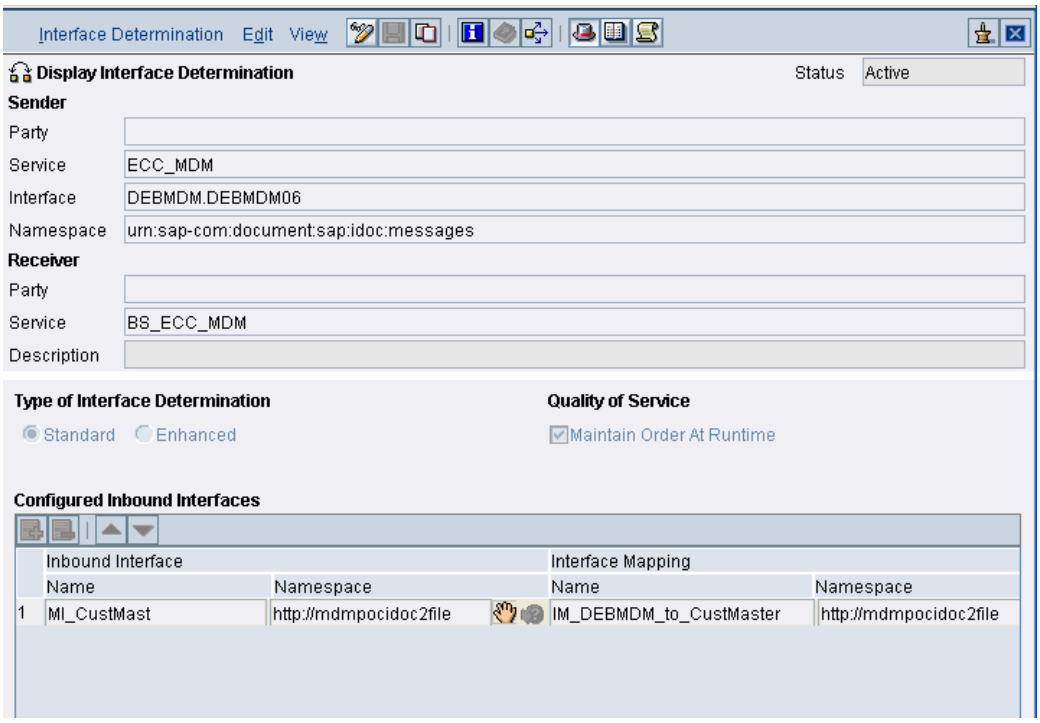
4.1. The next step is now to create the determinations and the agreements.
4.2. First is the Receiver Determination. Right click on *Receiver Determination* and click on *New*.
4.3. Choose the appropriate Sender and Receiver services, the outbound interface etc.
4.4. Save.

***Note**:* Sender Agreement not required as IDoc itself sits on the ABAP stack.



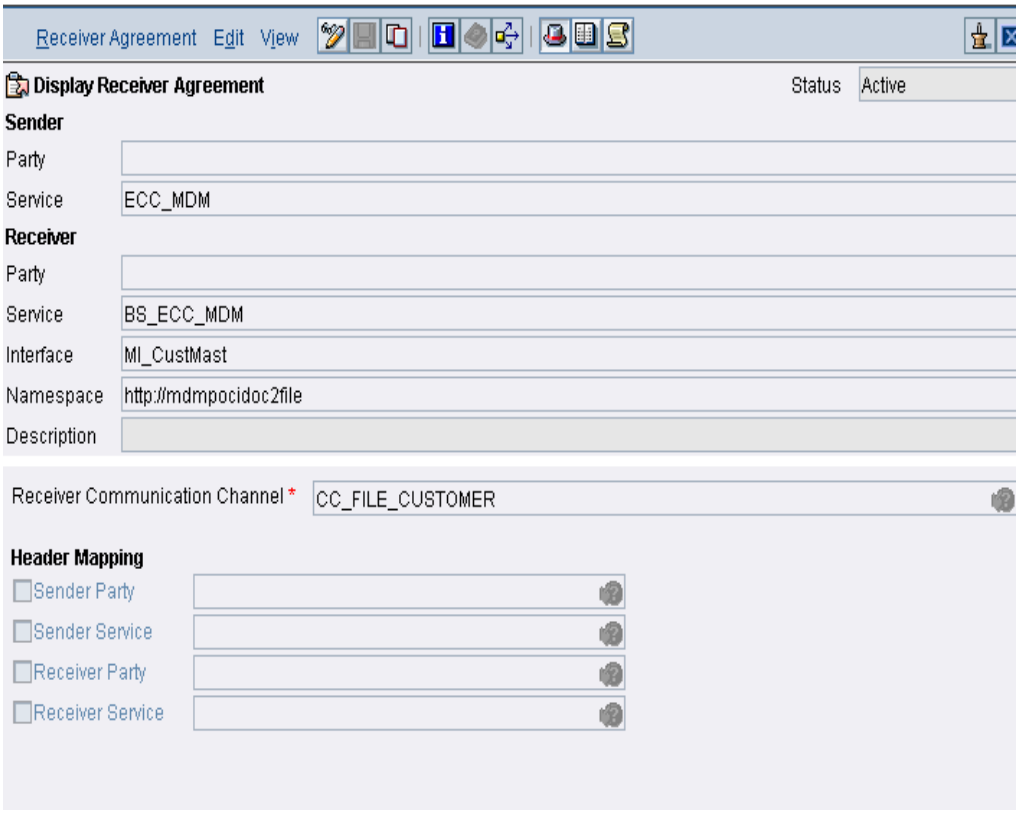
5. Creation of Interface Determination

5.1. Next is to create the Interface Determination.
5.2. Click on *Interface Determination->New.*
5.3. Again select the appropriate parameters.
5.4. Save.



6. Creation of Receiver Agreement

6.1. The last step is to create the Receiver Agreement.
6.2. Click on *Receiver Agreement->New.*
6.3. Select the appropriate values for the other fields. Also, select the Receiver Communication channel that will be used.



Glossary:

1. **Software Component:**

   The SWCV acts as a container for the development objects viz. data types, message types, message interfaces, message mappings etc. It has to be created in the SLD and then imported into the IR.

   Once the SWCV is imported, it can be assigned two things:

   I. Namespaces

   II.A connection to an existing SAP system for import of IDOC/RFC interfaces.

2. **Namespace:**

   A Namespace is a (globally) unique identifier for related Integration Repository objects. It is usually expressed as a URI or a URN.

3. **Data Type:**

   A Data type is the most basic entity to define the structure of XML elements.
   It defines the structures of the source and target messages.

4. **Message Type:**

   A Message Type comprises a data type that describes the structure of a message. For technical reasons, a data type is not sufficient to describe the instance of a message. In XML schema, data types are defined as abstract types that are not yet fixed to an element. We can only describe an instance of a message when we have specified a data type as an element type. Therefore, a Message Type defines the root element of a message.

5. **Message Interface:**
   The Message Interface is the highest-level representation of XML metadata.

   It is created using the Message Type. It is used to exchange messages between the systems. Depending on its usage, it can have a task of either sending the message (outbound) or receiving the message (inbound).

6. **Message Mapping:**

   When different interfaces are used in an Integration Scenario, we must provide a mapping to transform the structures of the interfaces. Mapping is done between the source and the target structures.

7. **Interface Mapping:**

   An Interface Mapping registers a pair of interfaces for use in a scenario, and specifies the message mapping (s) to be used. Mapping programs can be called sequentially.

8. **Configuration Scenario:**

   In order to structure the content of the Integration Directory more clearly, configuration objects are grouped using a configuration scenario.

9. **Business System:**

   A Business system is a logical system, which functions as sender or receiver within XI. It can be SAP system or a third-party system.

10. **Business Service:**

    A Business Service represents an abstract unit for addressing message senders and message receivers. A Business Service can be used to group interfaces.

11. **Communication Channel:**

    A communication channel defines type and configuration of the adapter to be used during inbound and outbound processing in the scenario. Here the Adapter specific identifiers like the Message protocol to be used, the Transport Protocol etc have to be given. In addition, there are also some other details that have to be specified like the name of the server, port etc.

    There are 2 types of Communication Channels: Sender and Receiver. The Sender Communication Channel picks up the file from the specified folder while the Receiver Communication Channel posts the file in the desired folder or posts IDoc in the specified ERP system.

12. **Sender Agreement:**

    A Sender Agreement is used during inbound processing. Here, we define how the message is to be transformed so that it can be processed by the Integration Engine.

13. **Receiver Determination:**

    A Receiver Determination is used to specify which receivers a particular message is to be sent to. There is also an option for defining conditions for forwarding the message to the receivers. Services or Systems are specified as receivers of messages.

14. **Interface Determination:**

    Interface Determination is used to specify a sender, an outbound interface, and a receiver, which inbound interface is to be used for receiving the message at the receiver. It also specifies which Interface Mapping from the Integration Repository is to be used for processing the message.

15. **Receiver Agreement:**

    A Receiver Agreement is used during outbound processing. Here, we define how the message is to be transformed so that it can be processed by a receiver.

Note: The Business Service and the Communication Channels are called Collaboration Profile objects, the two Agreements are Logical Routing objects while the two Determinations are Collaboration Agreement objects.

Summary

Through this article we have seen the various steps that need to be followed in PI to build an IDoc to XML scenario for exchange of messages between ERP and MDM systems.

Related Contents

https://www.sdn.sap.com/irj/scn/weblogs?blog=/pub/wlg/1819

https://www.sdn.sap.com/irj/scn/weblogs?blog=/pub/wlg/2923

https://help.sap.com

For more information, visit the Master Data Management homepage.

Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.