

# Java Web Dynpro: Create Sales Order Using Bapi

**Author:** Monica Radytia Chrismawati

**Created on:** 15 June 2013

**Applies to:**

SAP Netweaver Composition Environment 7.3

**Summary:**

In this tutorial, we will try to create sales order using BAPI in Java Web Dynpro. The final result will be like the screenshot below. We will try to bind data that comes from user input into BAPI. This tutorial will use **BAPI\_SALES\_CREATEFROMDAT2**, **BAPI\_TRANSACTION\_COMMIT**, and **BAPI\_TRANSACTION\_ROLLBACK**. The main concept about the use of BAPI in Java Web Dynpro is BAPI will be imported as model data that will be converted as java class which will be used in developing java code.

As your basic knowledge in SAP SD, sales order is formed from three big section: header level, item level, and schedule lines level. We will provide a form that will be filled up with header data, multiple items data that can be added and deleted, then will show the return log after user press the button to create sales order.

**CREATE SALES ORDER**

**Header Data**

Doc\_Type\_label: TA  
 Sales\_Org\_label: 1000  
 Distr\_Chain\_label: 10  
 Division\_label: 00  
 Partn\_Numb\_label: 1000  
 Purch\_No\_C\_label: PO NO

**Item Data**

Material\_label: T-ATA20  
 Target\_Qty\_label: 8  
 Target\_Qu\_label: ST

Add Delete

Item	Material	Target quantity	UoM
000010	T-ATA20	3	ST
000020	T-ATA20	8	ST

Create SO

Sales Document: 15446

Fid	Message	MsgType	Msg.No.	Message Variable	Message Variable	Message Variable	Msg.no	Log number	Message Variable	Lines in parameter	Message ID	Param. Name	System
	SALES_HEADER_IN HAS BEEN PROCESSED SUCCESSFULLY	S	233	VBAKKOM			000000			0	V4	SALES_HEADER_IN	T90CLN
	SALES_ITEM_IN HAS BEEN PROCESSED SUCCESSFULLY	S	233	VBAPKOM		000010	000000			1	V4	SALES_ITEM_IN	T90CLN
	SALES_ITEM_IN HAS BEEN PROCESSED SUCCESSFULLY	S	233	VBAPKOM		000020	000000			2	V4	SALES_ITEM_IN	T90CLN
	SALES_SCHEDULES_IN HAS BEEN PROCESSED SUCCESSFULLY	S	233	VBEPKOM			000000			1	V4	SALES_SCHEDULES_IN	T90CLN
	SALES_SCHEDULES_IN HAS BEEN PROCESSED SUCCESSFULLY	S	233	VBEPKOM			000000			3	V4	SALES_SCHEDULES_IN	T90CLN

**Tools:**

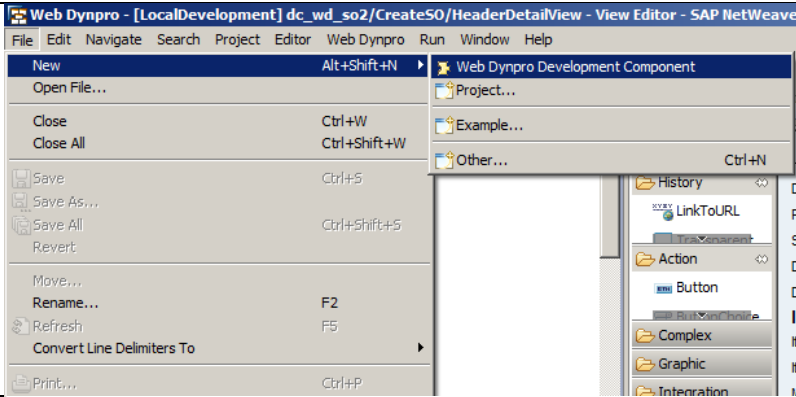
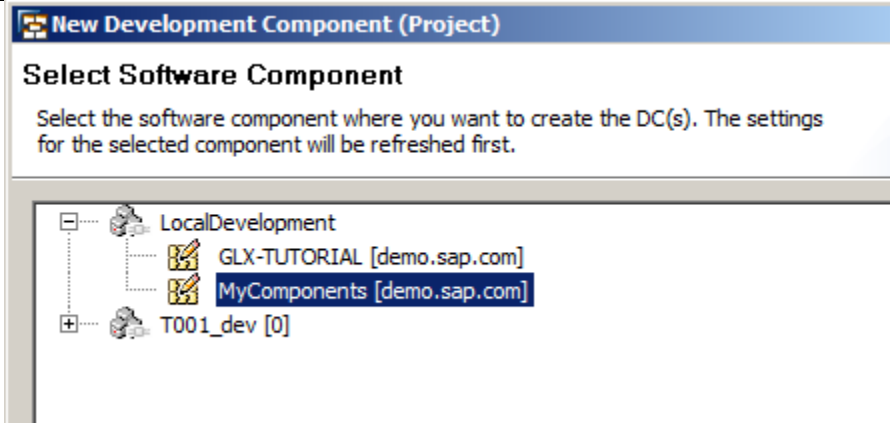
- ☐ SAP NetWeaver Developer Studio SAP Netweaver 7.3 SP00 PAT0000.

**Prerequisite:**

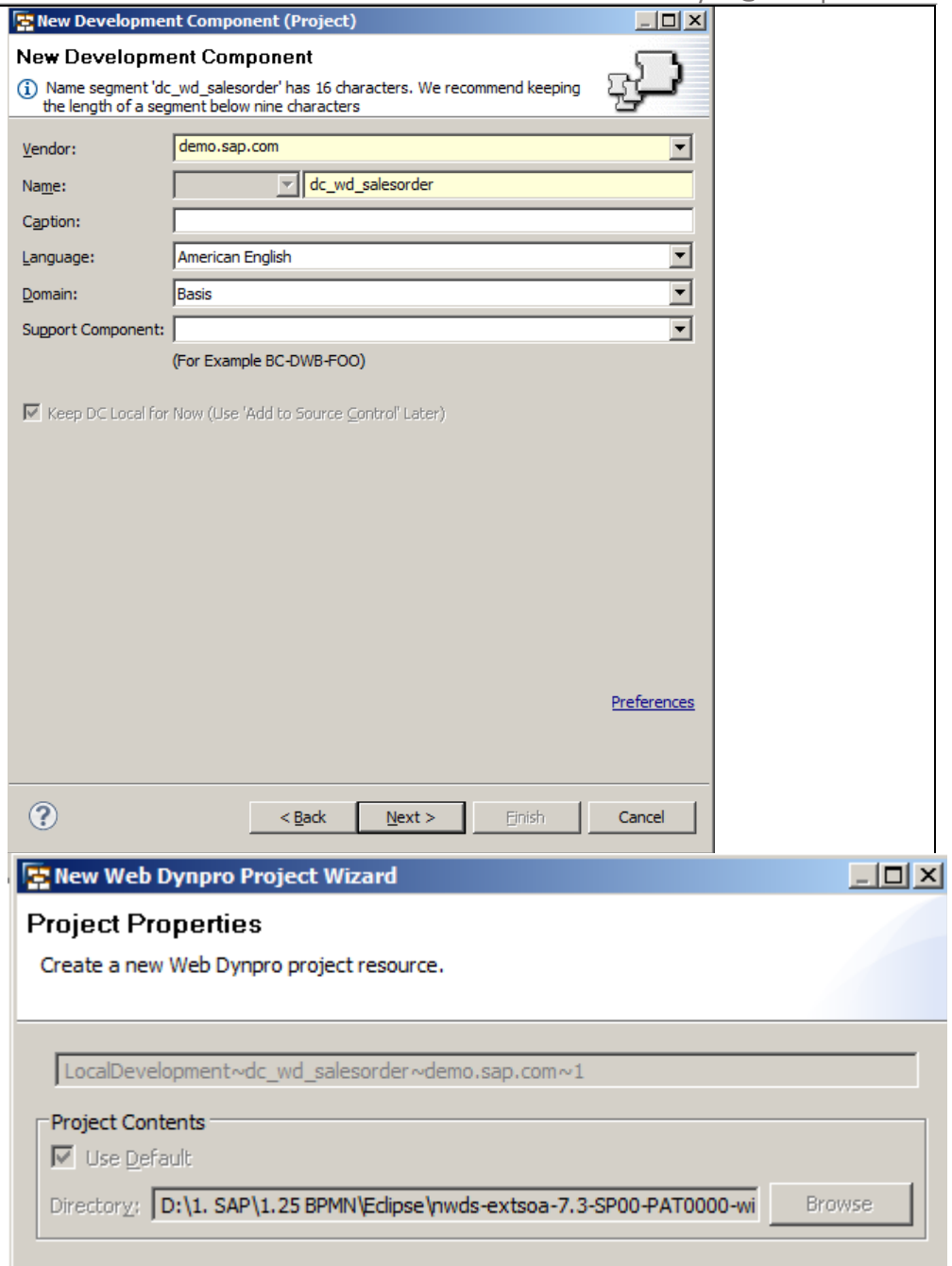
- ☐ SAP NetWeaver Developer Studio is installed in your computer
- ☐ SAP J2EE Engine is up.
- ☐ SAP As Java in NWDS already configured.
- ☐ R/3 Configuration is already configured.
- ☐ You have access to a remote SAP back-end system (SAP R/3).
- ☐ The SAP System Landscape Directory (SLD) and the SLD bridge are configured and running.

**Knowledge:**

- ☐ You have some experience with Java Web Dynpro applications.
- ☐ You have experience in Java programming.
- ☐ You are familiar with the use of Function Modules and BAPIs.

<b>STEP 1. CREATE A NEW DEVELOPMENT COMPONENT</b>		
1.	Start SAP NetWeaver Developer Studio (NWDS). <i>File -&gt; New -&gt; Web Dynpro Development Component.</i>	
2.	Select the appropriate <i>Software Component</i> then click <i>Next</i> button.	

3. A new window to create *Development Component* appear. Type "dc\_wd\_salesorder" as name of the new development component. Click *Next*.



The screenshot displays two SAP development component creation windows. The top window, titled "New Development Component (Project)", contains the following fields: Vendor (demo.sap.com), Name (dc\_wd\_salesorder), Caption, Language (American English), Domain (Basis), and Support Component. A message at the top states: "Name segment 'dc\_wd\_salesorder' has 16 characters. We recommend keeping the length of a segment below nine characters". A checkbox "Keep DC Local for Now (Use 'Add to Source Control' Later)" is checked. A "Preferences" link is at the bottom right. The bottom window, titled "New Web Dynpro Project Wizard", shows "Project Properties" with the text "Create a new Web Dynpro project resource." and a text field containing "LocalDevelopment~dc\_wd\_salesorder~demo.sap.com~1". Under "Project Contents", the "Use Default" checkbox is checked, and the "Directory" field shows "D:\1. SAP\1.25 BPMN\Eclipse\nwds-extsoa-7.3-SP00-PAT0000-wi" with a "Browse" button.

**New Development Component (Project)**

**New Development Component**

① Name segment 'dc\_wd\_salesorder' has 16 characters. We recommend keeping the length of a segment below nine characters

Vendor: demo.sap.com

Name: dc\_wd\_salesorder

Caption:

Language: American English

Domain: Basis

Support Component:

(For Example BC-DWB-FOO)

☒ Keep DC Local for Now (Use 'Add to Source Control' Later)

[Preferences](#)

< Back Next > Finish Cancel

**New Web Dynpro Project Wizard**

**Project Properties**

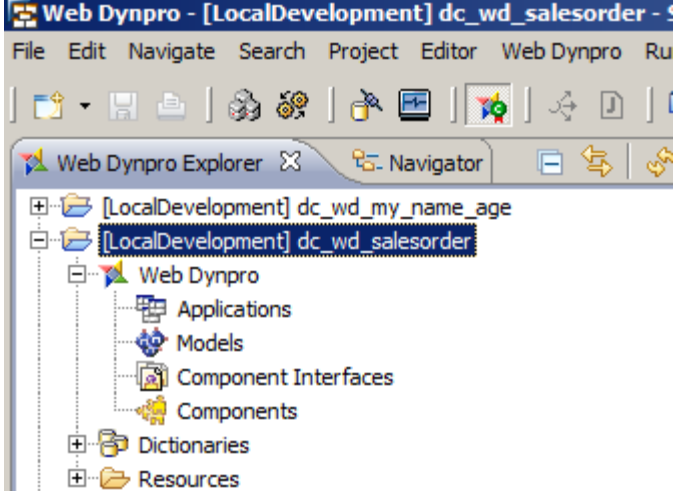
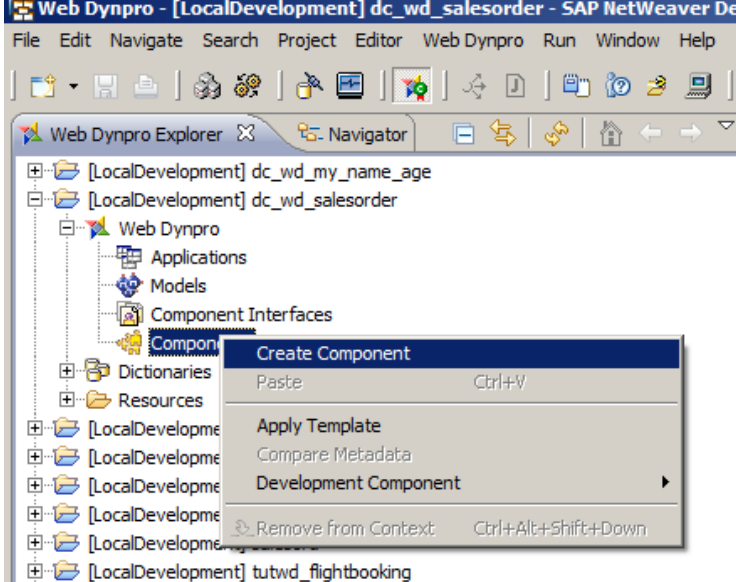
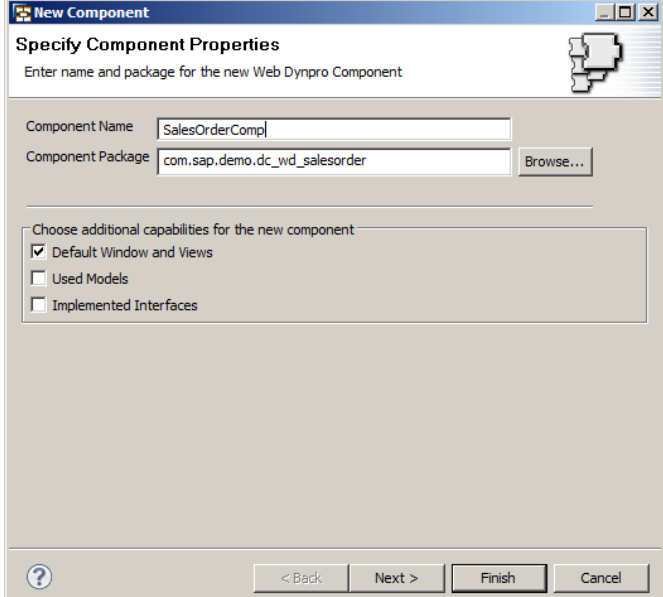
Create a new Web Dynpro project resource.

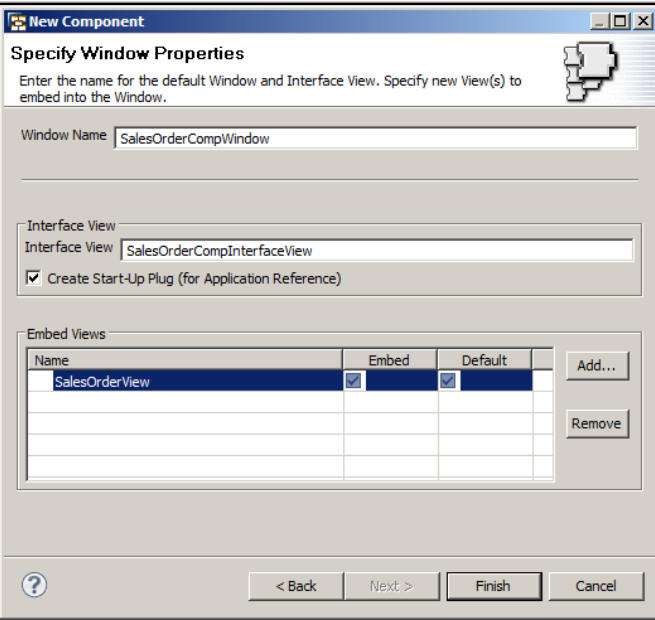
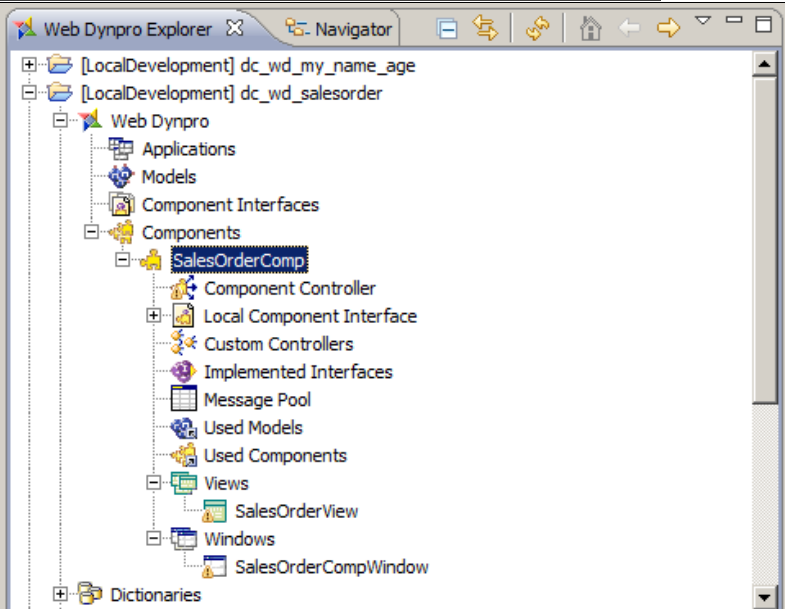
LocalDevelopment~dc\_wd\_salesorder~demo.sap.com~1

**Project Contents**

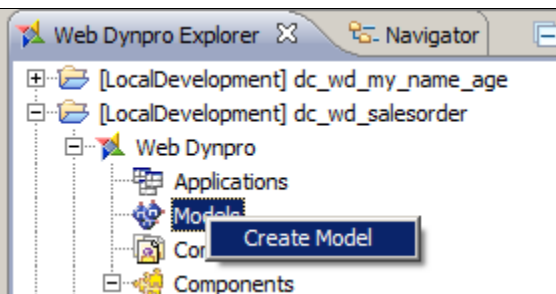
☒ Use Default

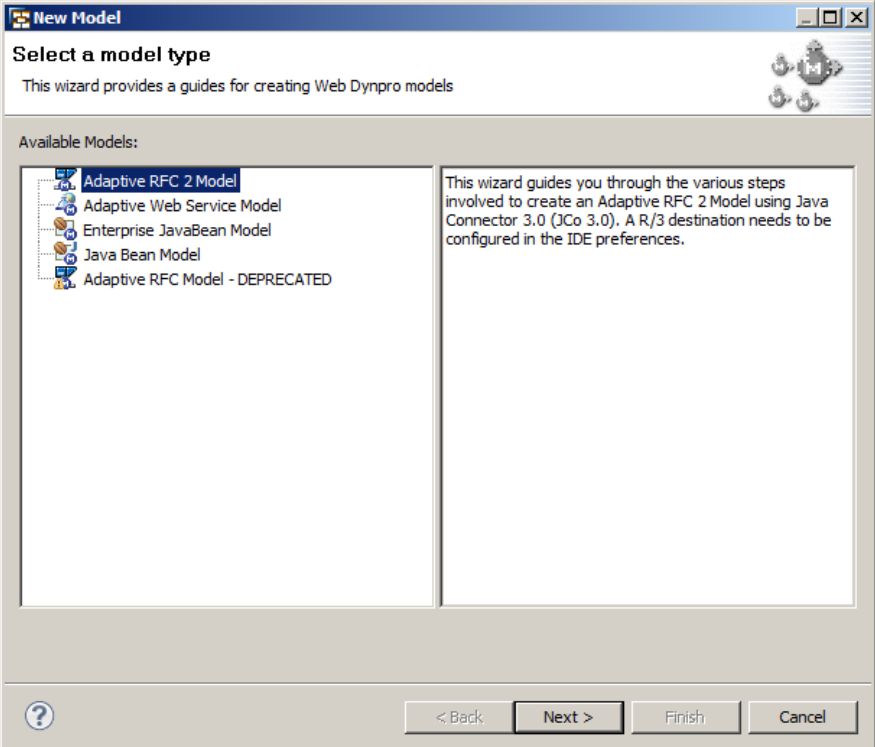
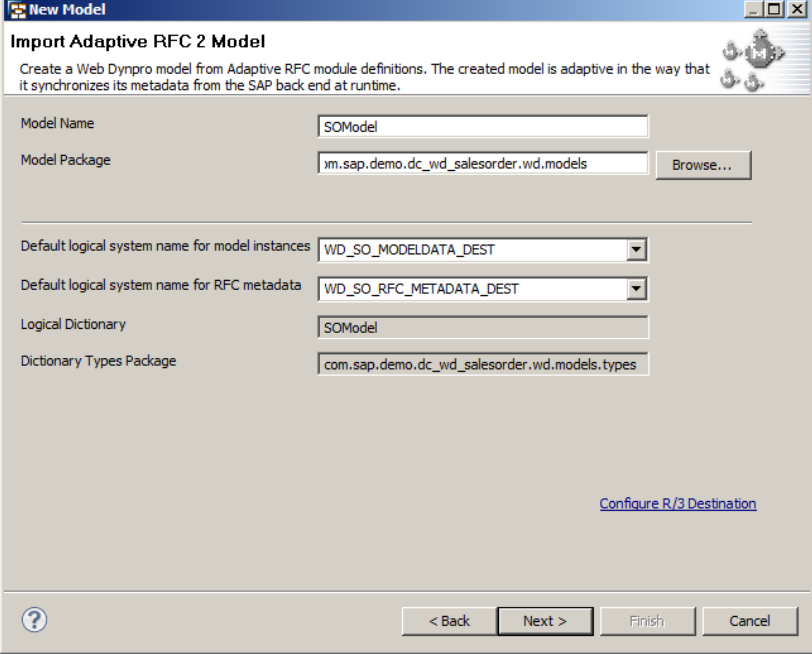
Directory: D:\1. SAP\1.25 BPMN\Eclipse\nwds-extsoa-7.3-SP00-PAT0000-wi Browse

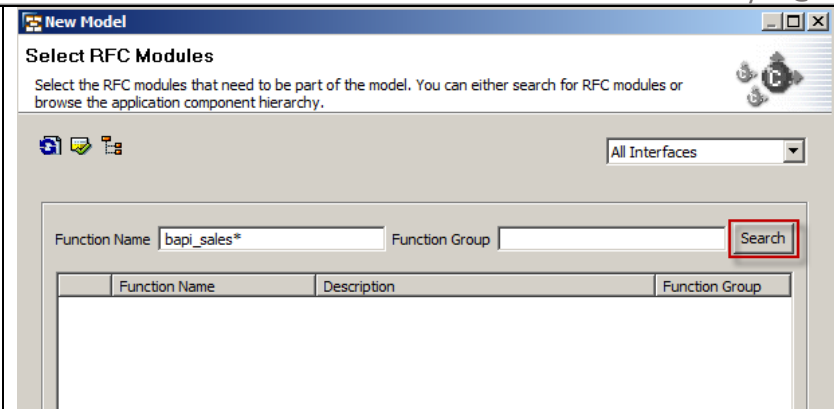
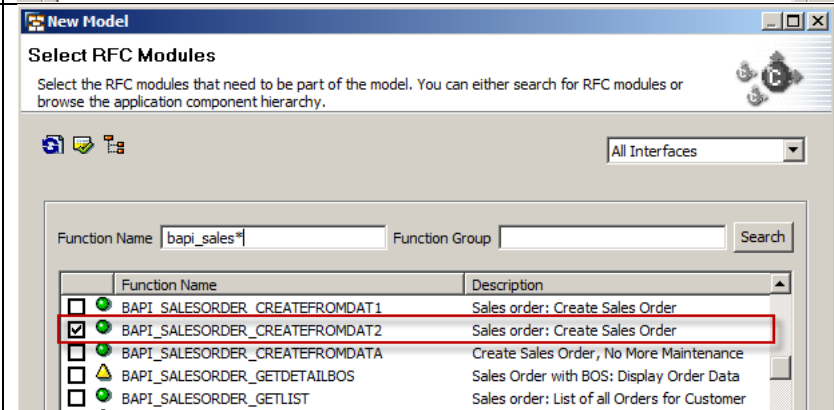
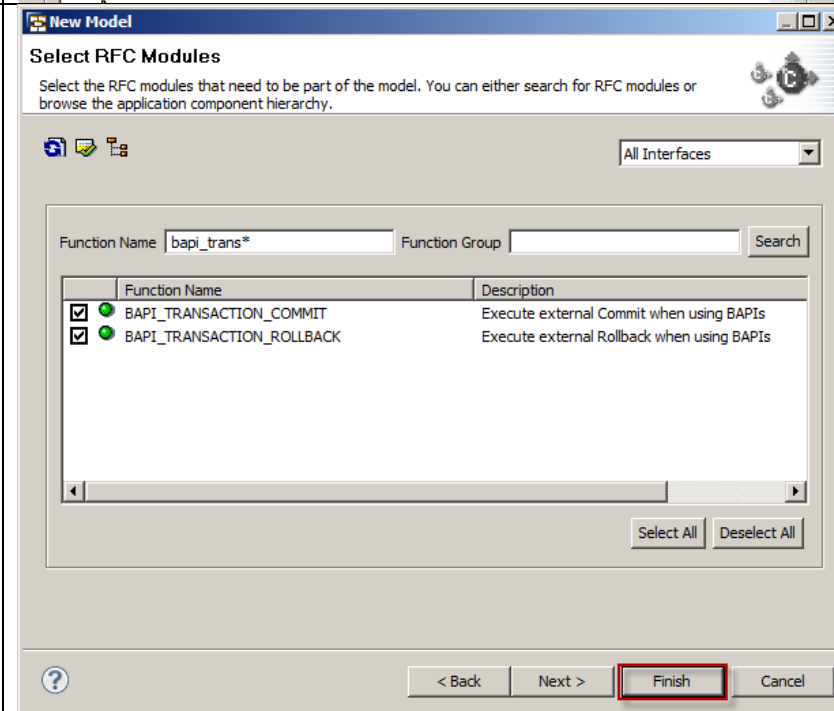
4.	A new web dynpro development component is created. An empty structure of this dc will be shown like screenshot beside.	
<b>STEP 2. CREATE COMPONENT</b>		
5.	Click on node <i>Component</i> then right click on it. Choose <i>Create Component</i> .	
6.	Type "SalesOrderComp" in <i>Component Name</i> field. Then choose the appropriate <i>Component Package</i> . Thick on <i>Default Windows and Views</i> only, since we will used model later.	

7.	A new window appear. If there is no revise from suggested name of window, interface, and view, continue by click <i>Finish</i> .	
8.	The structure of our dc after created a component will be look like this screenshot.	

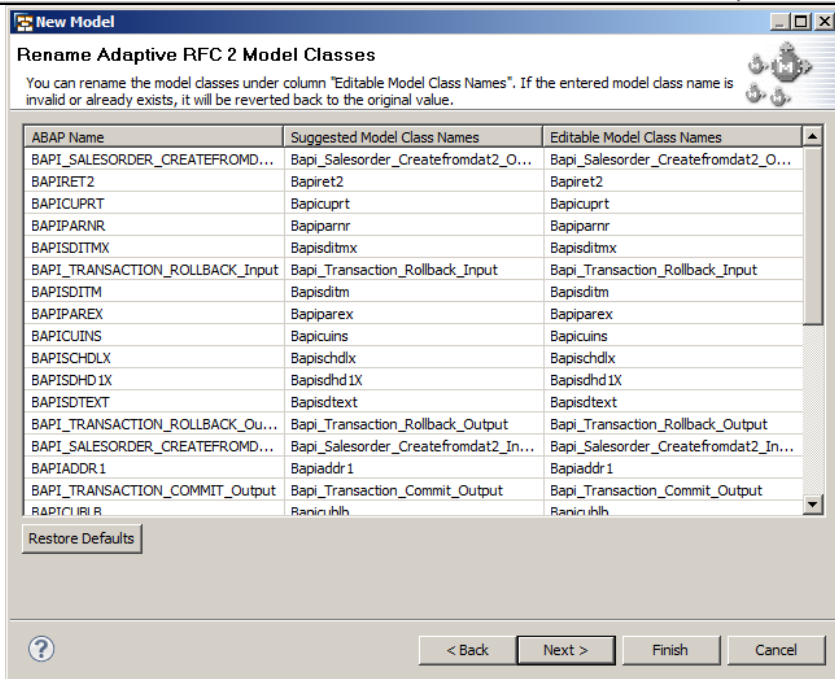
### STEP 3. IMPORTING BAPI AS MODEL

9.	Right click on <i>Models</i> > <i>Create Model</i> .	
----	------------------------------------------------------	--------------------------------------------------------------------------------------

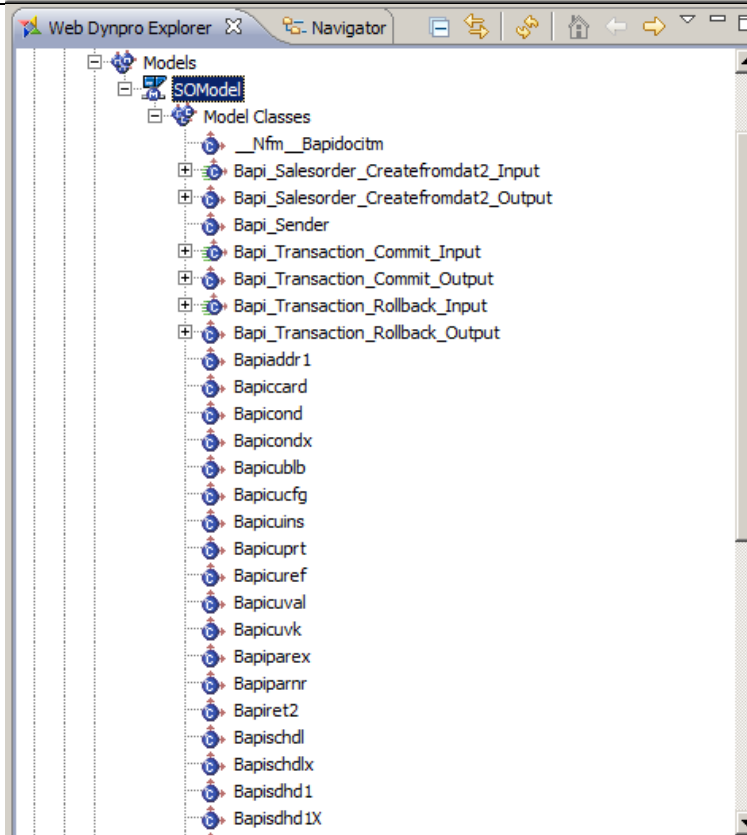
10.	Click on <i>Adaptive RFC 2 Model</i> > <i>Next</i> .	
11.	<p>Type "SOModel" in Model Name. Browse appropriate model package, or rename the suggested name.</p> <p>Rename the Default logical name for model instance: "WD_SO_MODELDATA_DEST" and Default logical system name for RFC metadata: "WD_SO_RFC_METAD ATA_DEST"</p> <p>Click Next.</p>	

12.	In new window appear, type in Function Name for the 3 BAPIs that will be used in this tutorial: <ul style="list-style-type: none"><li>- BAPI_SALES_CREATEFROMDAT2</li><li>- BAPI_TRANSACTION_COMMIT</li><li>- BAPI_TRANSACTION_ROLLBACK</li></ul>	
13.	Thick on three BAPIs that will we use.	
14.	Click Finish.	

15. If you need rename, on next screen appear you can do that, but in this tutorial, we will remain the name of model as default. Click Next.

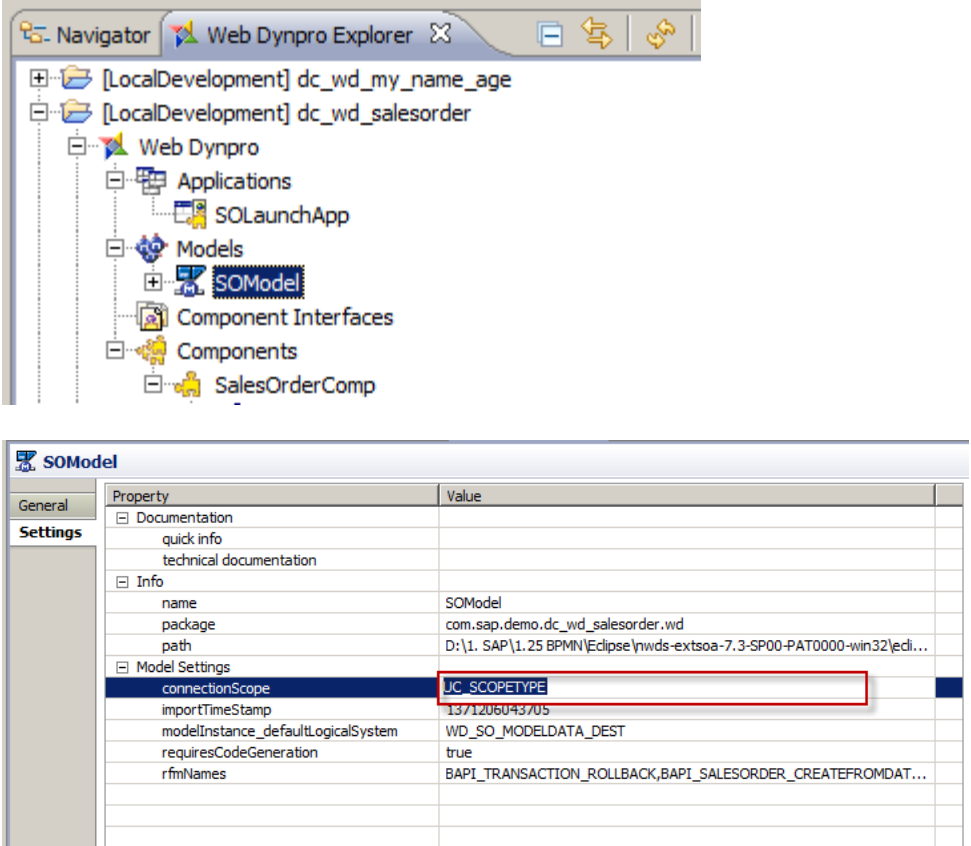
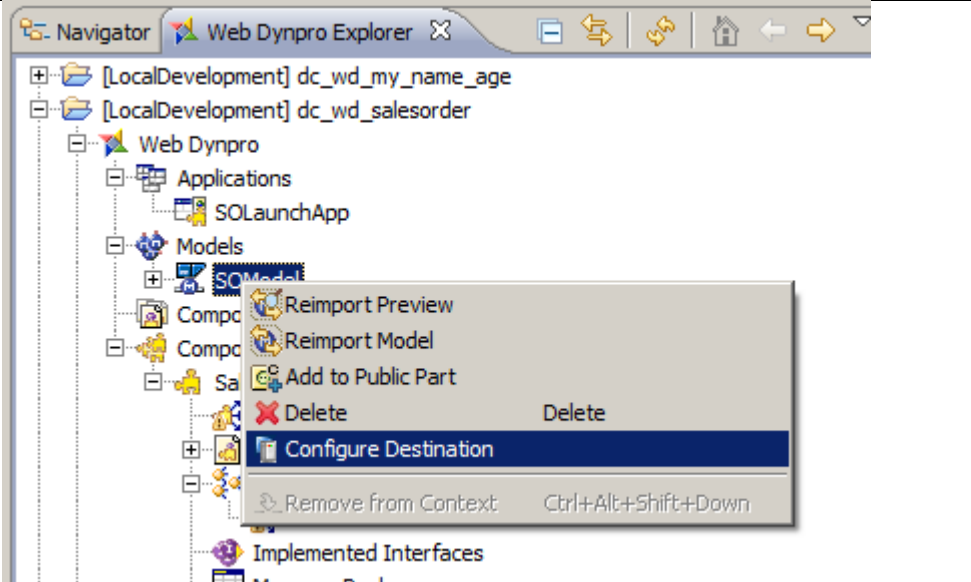


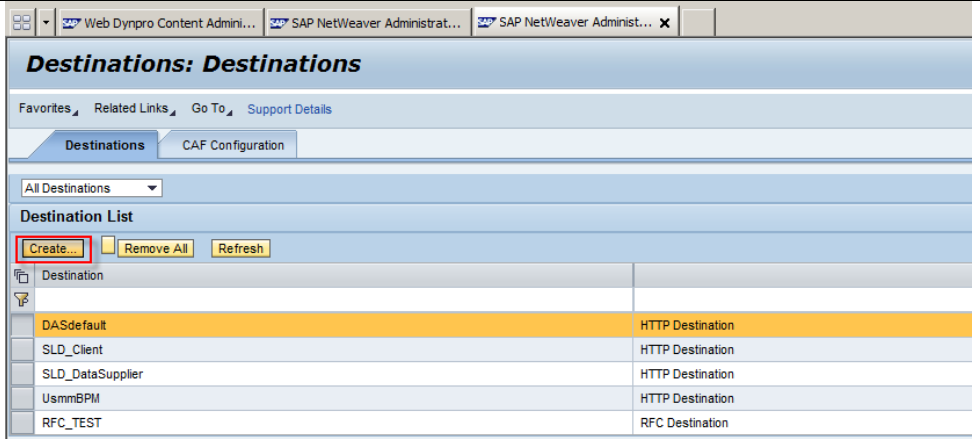
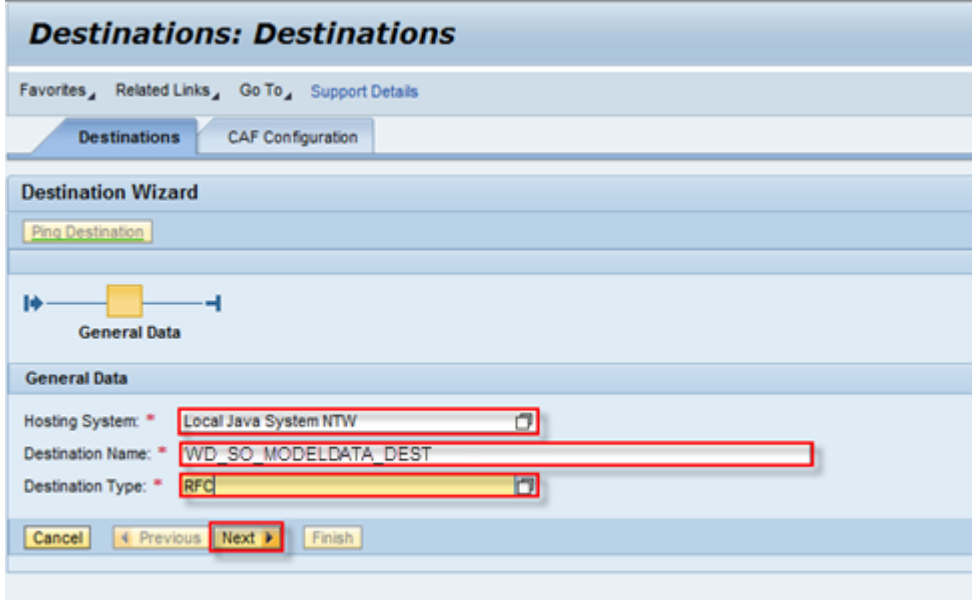
16. If success, then tree structure of SOModel will appear as screenshot beside.

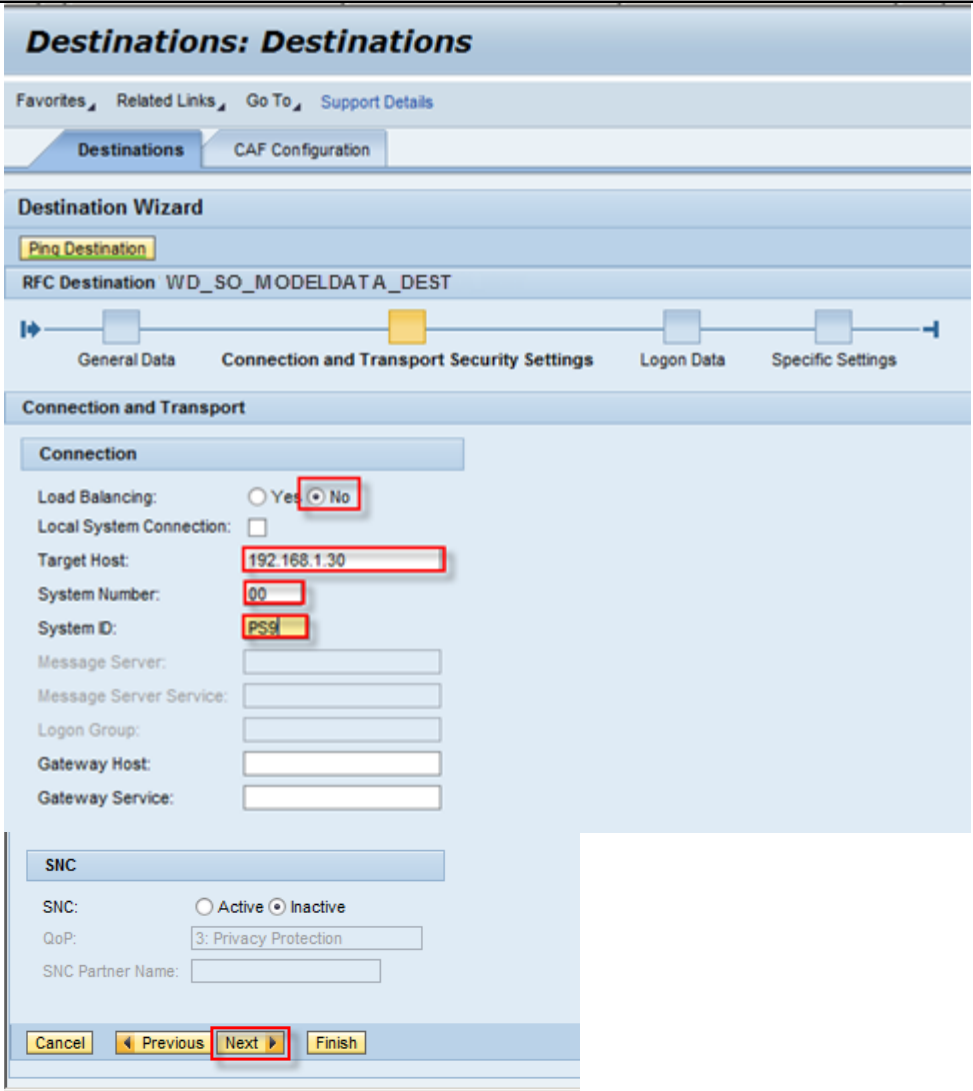
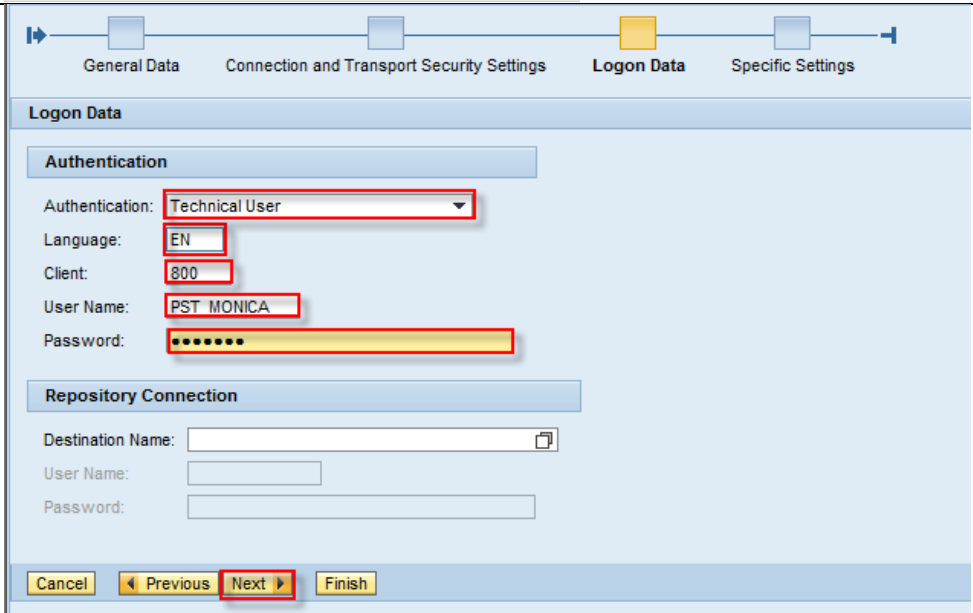


## STEP 4. CONFIGURE MODEL DESTINATION



17.	<p>Make sure your model's connection type is "UC_SCOPETYPE".</p> <p>Double click on model you have.</p>	 <p>The screenshot shows the Web Dynpro Explorer interface. In the Navigator, the 'SOModel' is selected under the 'Models' folder. The 'SOModel' properties are displayed in the main area. The 'connectionScope' property is highlighted with a red box and set to 'UC_SCOPETYPE'.</p> <table><tr><th>Property</th><th>Value</th></tr><tr><td>Documentation</td><td></td></tr><tr><td>quick info</td><td></td></tr><tr><td>technical documentation</td><td></td></tr><tr><td>Info</td><td></td></tr><tr><td>name</td><td>SOModel</td></tr><tr><td>package</td><td>com.sap.demo.dc_wd_salesorder.wd</td></tr><tr><td>path</td><td>D:\1. SAP\1.25 BPMN\Eclipse\nwds-extsoa-7.3-SP00-PAT0000-win32\edl...</td></tr><tr><td>Model Settings</td><td></td></tr><tr><td>connectionScope</td><td>UC_SCOPETYPE</td></tr><tr><td>importTimeStamp</td><td>13/12/06 04:37:05</td></tr><tr><td>modelInstance_defaultLogicalSystem</td><td>WD_SO_MODELDATA_DEST</td></tr><tr><td>requiresCodeGeneration</td><td>true</td></tr><tr><td>rftNames</td><td>BAPI_TRANSACTION_ROLLBACK,BAPI_SALESORDER_CREATEFROMDAT...</td></tr></table>	Property	Value	Documentation		quick info		technical documentation		Info		name	SOModel	package	com.sap.demo.dc_wd_salesorder.wd	path	D:\1. SAP\1.25 BPMN\Eclipse\nwds-extsoa-7.3-SP00-PAT0000-win32\edl...	Model Settings		connectionScope	UC_SCOPETYPE	importTimeStamp	13/12/06 04:37:05	modelInstance_defaultLogicalSystem	WD_SO_MODELDATA_DEST	requiresCodeGeneration	true	rftNames	BAPI_TRANSACTION_ROLLBACK,BAPI_SALESORDER_CREATEFROMDAT...
Property	Value																													
Documentation																														
quick info																														
technical documentation																														
Info																														
name	SOModel																													
package	com.sap.demo.dc_wd_salesorder.wd																													
path	D:\1. SAP\1.25 BPMN\Eclipse\nwds-extsoa-7.3-SP00-PAT0000-win32\edl...																													
Model Settings																														
connectionScope	UC_SCOPETYPE																													
importTimeStamp	13/12/06 04:37:05																													
modelInstance_defaultLogicalSystem	WD_SO_MODELDATA_DEST																													
requiresCodeGeneration	true																													
rftNames	BAPI_TRANSACTION_ROLLBACK,BAPI_SALESORDER_CREATEFROMDAT...																													
18.	<p>Right click on model, <i>SOModel &gt; Configure Destination</i>.</p>	 <p>The screenshot shows the Web Dynpro Explorer interface. The 'SOModel' is selected in the Navigator. A right-click context menu is open, and the 'Configure Destination' option is highlighted.</p>																												

19.	Click Create.	 <p>The screenshot shows the 'Destinations: Destinations' page in SAP NetWeaver. It includes a 'Destination List' table with columns for 'Destination' and 'Type'. The 'Create...' button is highlighted with a red box.</p> <table><tr><th>Destination</th><th>Type</th></tr><tr><td>DASdefault</td><td>HTTP Destination</td></tr><tr><td>SLD_Client</td><td>HTTP Destination</td></tr><tr><td>SLD_DataSupplier</td><td>HTTP Destination</td></tr><tr><td>UsmmBPM</td><td>HTTP Destination</td></tr><tr><td>RFC_TEST</td><td>RFC Destination</td></tr></table>	Destination	Type	DASdefault	HTTP Destination	SLD_Client	HTTP Destination	SLD_DataSupplier	HTTP Destination	UsmmBPM	HTTP Destination	RFC_TEST	RFC Destination
Destination	Type													
DASdefault	HTTP Destination													
SLD_Client	HTTP Destination													
SLD_DataSupplier	HTTP Destination													
UsmmBPM	HTTP Destination													
RFC_TEST	RFC Destination													
20.	Fill the Destination with your RFC Application Data First. Click Next.	 <p>The screenshot shows the 'Destination Wizard' in SAP NetWeaver, specifically the 'General Data' step. The 'Next' button is highlighted with a red box.</p> <p>General Data</p> <p>Hosting System: Local Java System NTW</p> <p>Destination Name: WD_SO_MODELDATA_DEST</p> <p>Destination Type: RFC</p> <p>Buttons: Cancel, Previous, Next, Finish</p>												

<p>21.</p> <p>Fill up the form with target host of your SAP Backend.</p> <p>Click Next.</p>	 <p>The screenshot shows the 'Destinations: Destinations' window with the 'Destination Wizard' tab selected. The 'RFC Destination' is 'WD_SO_MODELDATA_DEST'. The 'Connection and Transport Security Settings' step is active, indicated by a yellow square on the progress bar. The 'Connection' section contains the following fields: 'Load Balancing' (radio buttons for Yes and No, with 'No' selected and highlighted), 'Local System Connection' (checkbox, unchecked), 'Target Host' (text box with '192.168.1.30' highlighted), 'System Number' (text box with '00' highlighted), 'System ID' (text box with 'PS9' highlighted), 'Message Server' (text box), 'Message Server Service' (text box), 'Logon Group' (text box), 'Gateway Host' (text box), and 'Gateway Service' (text box). The 'SNC' section contains: 'SNC' (radio buttons for Active and Inactive, with 'Inactive' selected and highlighted), 'QoP' (text box with '3: Privacy Protection'), and 'SNC Partner Name' (text box). At the bottom are buttons for 'Cancel', 'Previous', 'Next' (highlighted with a red box), and 'Finish'.</p>
<p>22.</p> <p>Fill with your username that will you use to login in SAP Backend system.</p> <p>Click Next.</p>	 <p>The screenshot shows the 'Destinations: Destinations' window with the 'Logon Data' step active, indicated by a yellow square on the progress bar. The 'Logon Data' section contains the following fields: 'Authentication' (dropdown menu with 'Technical User' selected and highlighted), 'Language' (text box with 'EN' highlighted), 'Client' (text box with '800' highlighted), 'User Name' (text box with 'PST MONICA' highlighted), and 'Password' (password field with dots, highlighted). The 'Repository Connection' section contains: 'Destination Name' (text box), 'User Name' (text box), and 'Password' (text box). At the bottom are buttons for 'Cancel', 'Previous', 'Next' (highlighted with a red box), and 'Finish'.</p>

23. Leave default.  
Click Finish.

General Data   Connection and Transport Security Settings   Logon Data   **Specific Settings**

**Specific Data**

**Pool Settings**

Pooled Connection Mode: ☒

Max. Connections:

Pool Size:

Max. Wait Time in ms:

**Advanced Settings**

SAP Router String:

RFC Trace: ☐

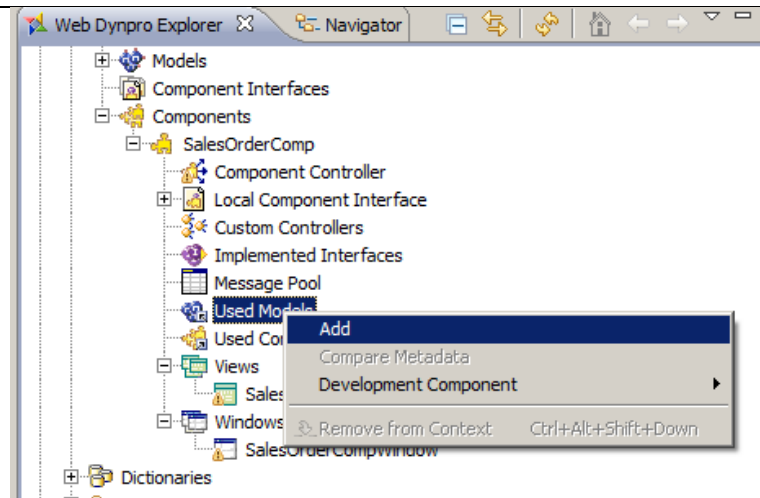
CPIC Trace Level:

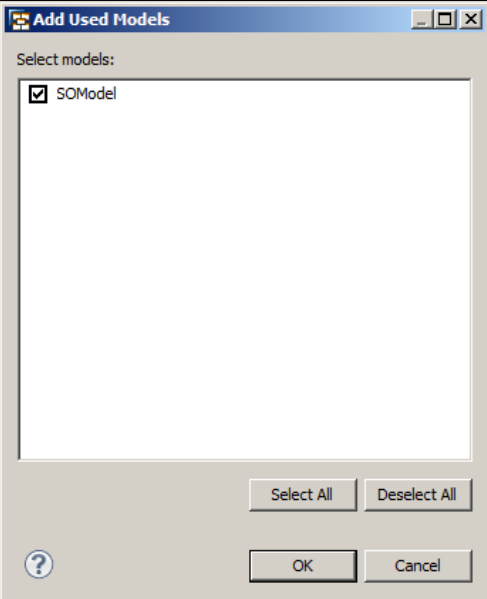
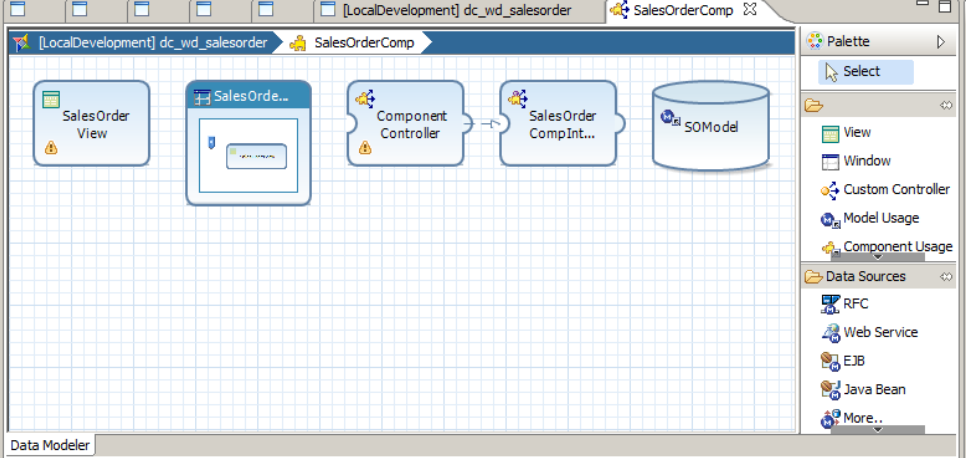
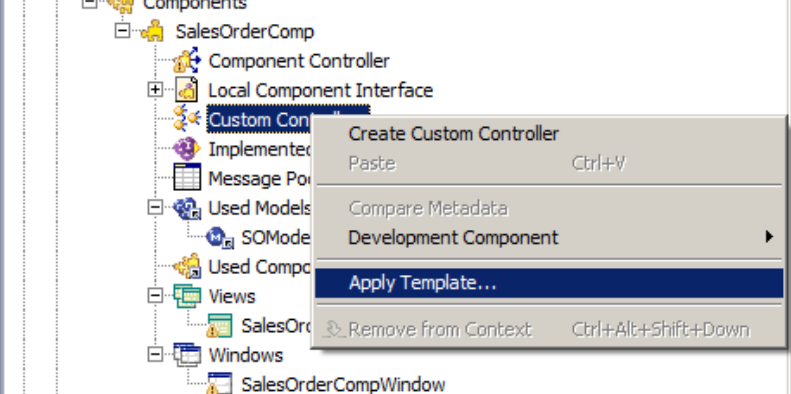
Code Page:

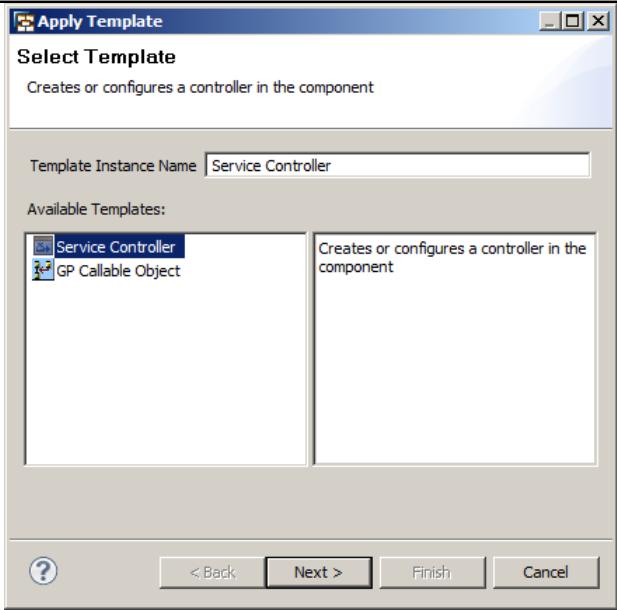
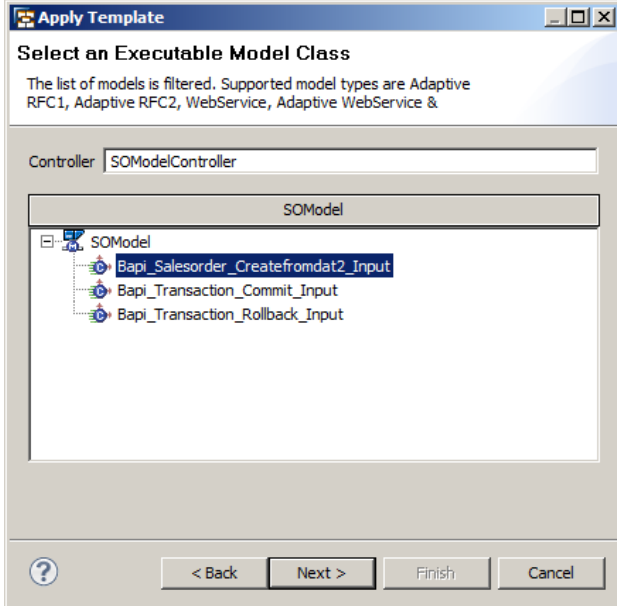
Cancel   Previous   Next   **Finish**

## STEP 5. USED MODEL IN COMPONENTS

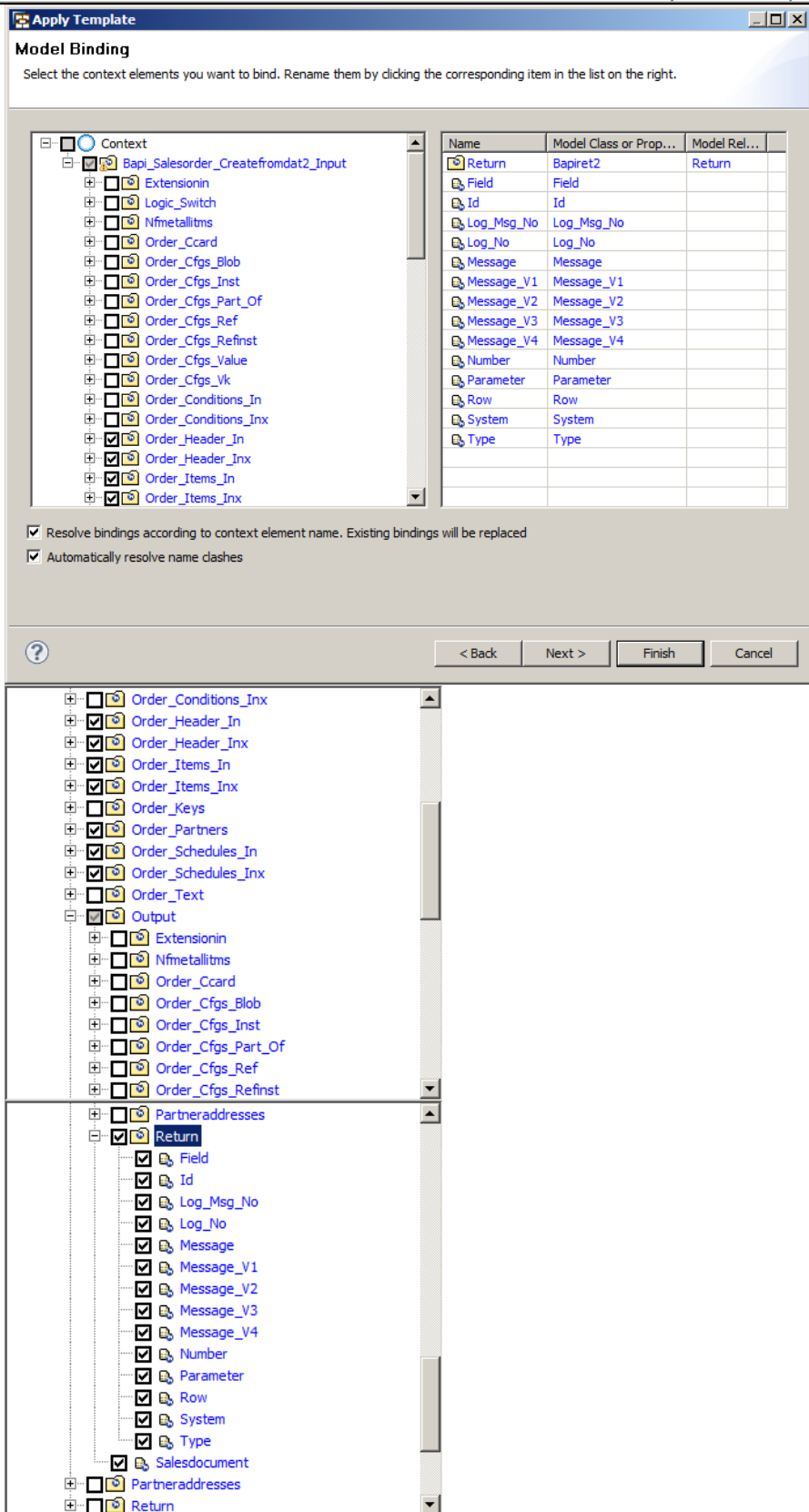
24. Expand the structure  
of the tree  
Components.  
Right click on *Used  
Models* > Add.

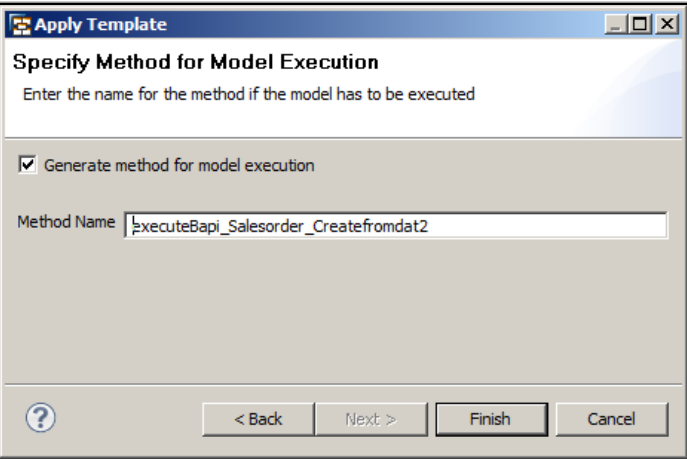
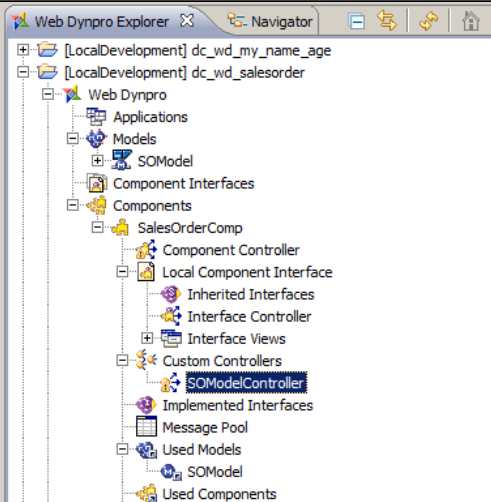


25.	On pop up window appear, select models. Click OK.	
26.	Open the Data Modeler of the components. Here we can see that model already used in our SalesOrderComp.	
<b>STEP 6. CREATE CUSTOM CONTROLLER</b>		
27.	Right click on <i>Custom Controller</i> > <i>Apply Template</i> .	

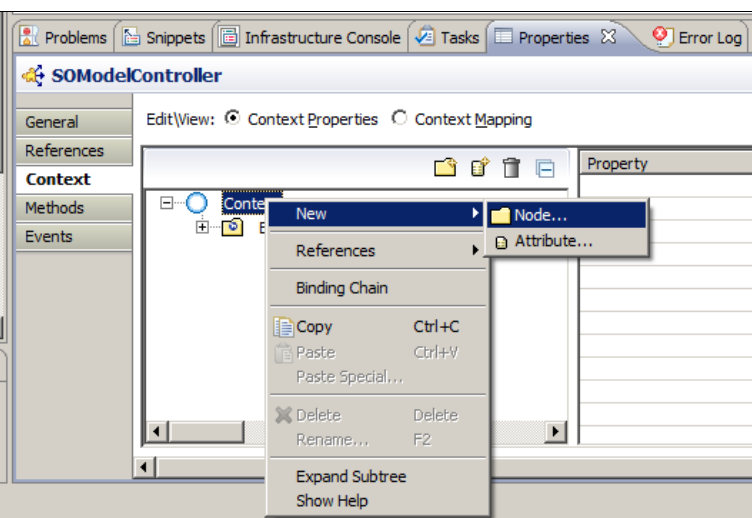
28.	Choose Service Controller. Click Next.	
29.	Choose our first model that will we use. Click Next.	

30. Structure of BAPI\_SALES\_CREATE\_FROMDAT2 appear. Here we only use:  
**Order\_Header\_In,**  
**Order\_Header\_Inx,**  
**Order\_Items\_In,**  
**Order\_Items\_Inx,**  
**Order\_Partners,**  
**Order\_Schedules\_In,**  
**Order\_Schedules\_Inx,**  
**Output.**  
 Click Finish.

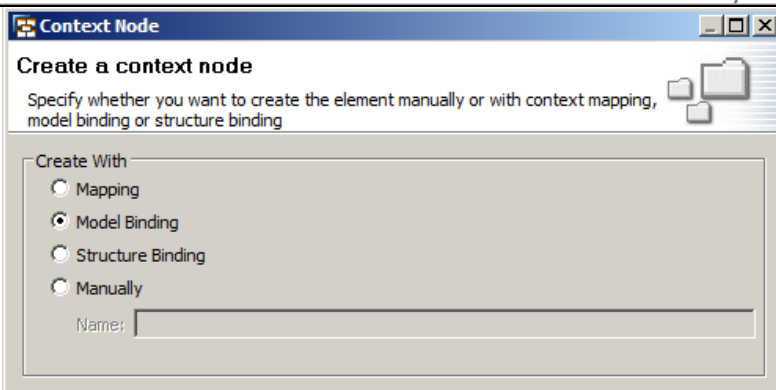
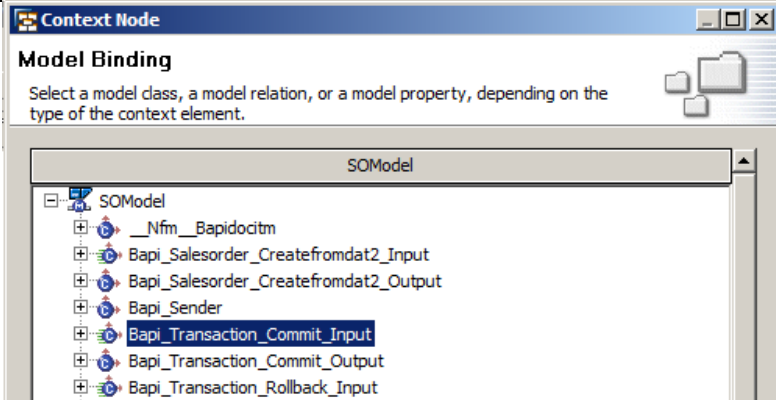
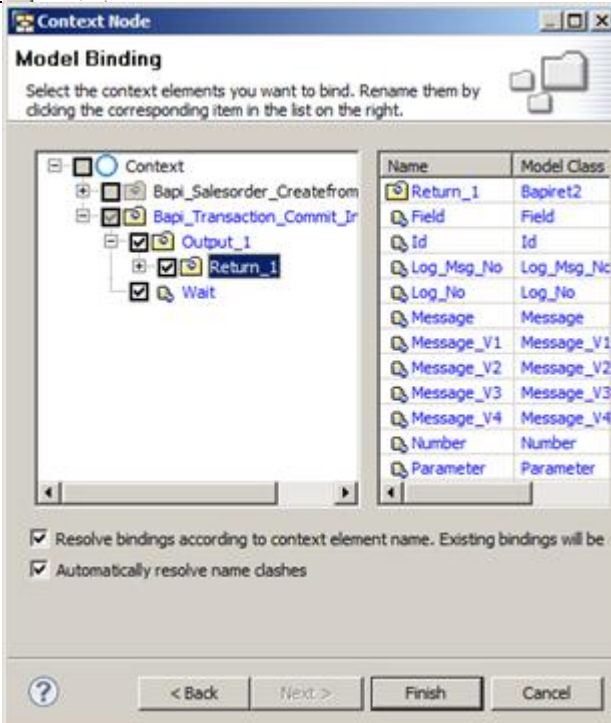


31.	On Method Name rename the method if you will. If not leave default. Click Finish.	
32.	Custom Controller <i>SOModelController</i> is created. The model <b>Bapi_Salesorder_Createfromdat2_Input</b> also automatically bind into custom controller's context.	

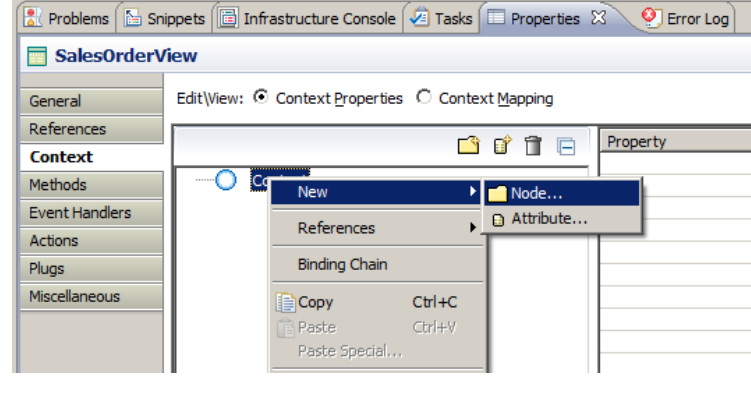
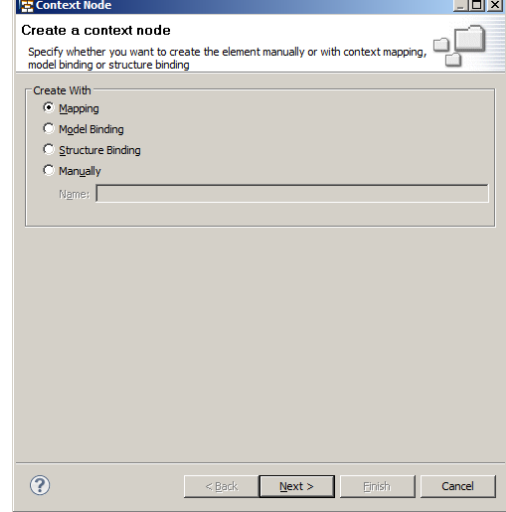
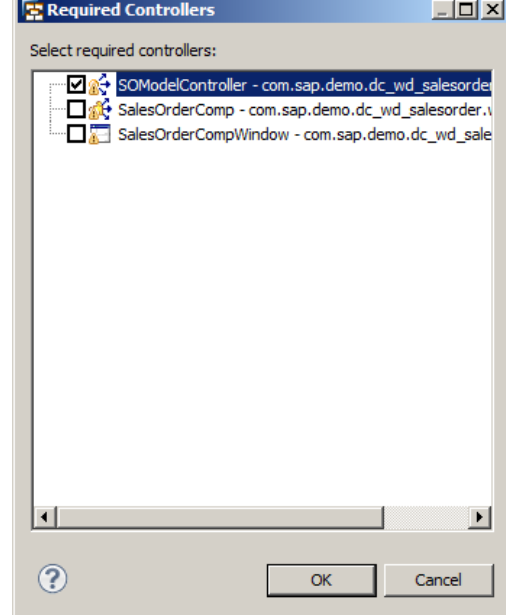
## STEP 7. ADD CONTEXT USING MODEL BINDING IN CUSTOM CONTROLLER

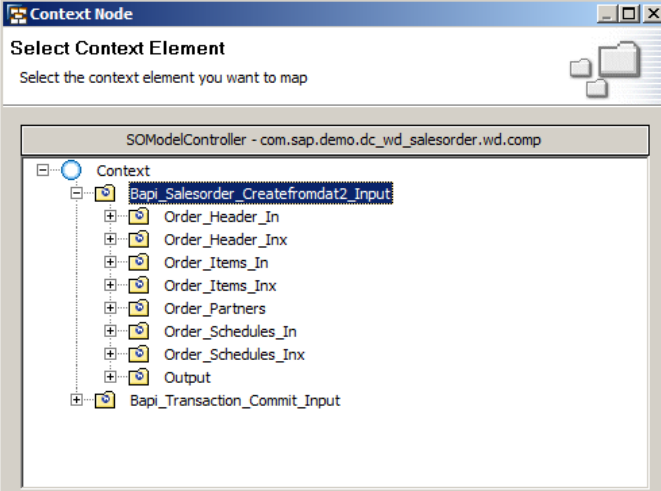
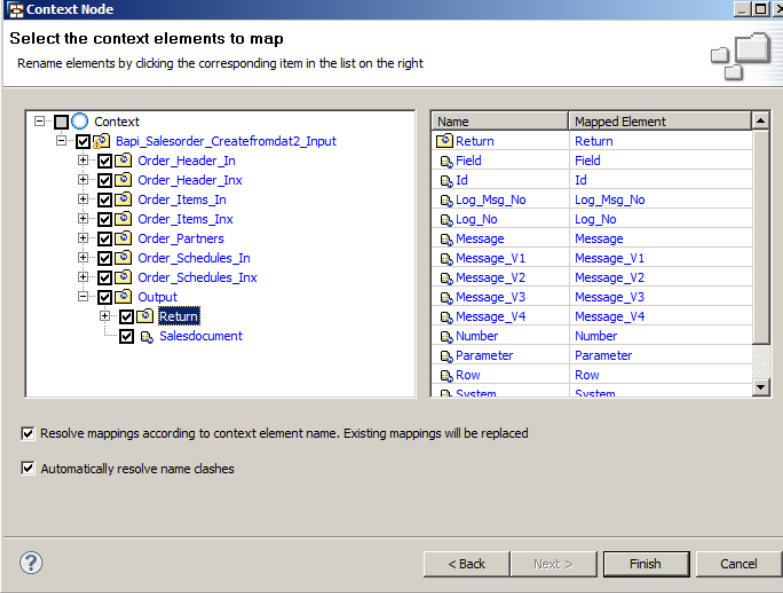
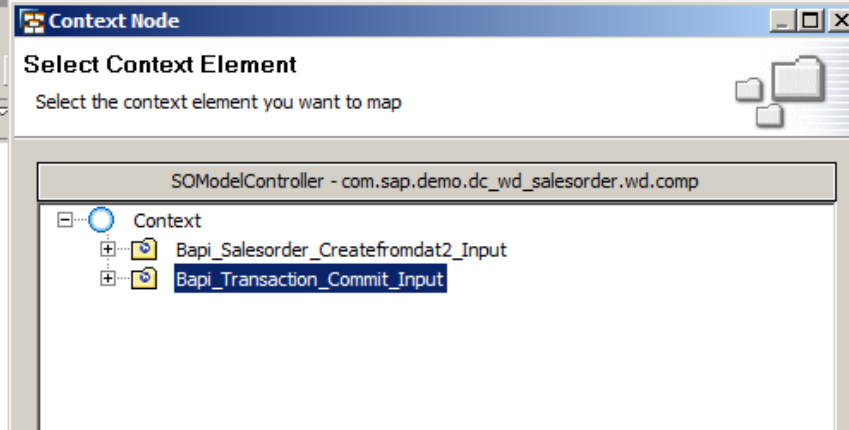
33.	Add another model of <b>Bapi_Transaction_Commit_Input</b> . Double click in custom controller. Open tab Context. Right click on <i>Context</i> > <i>New</i> > <i>Node</i> .	
-----	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------



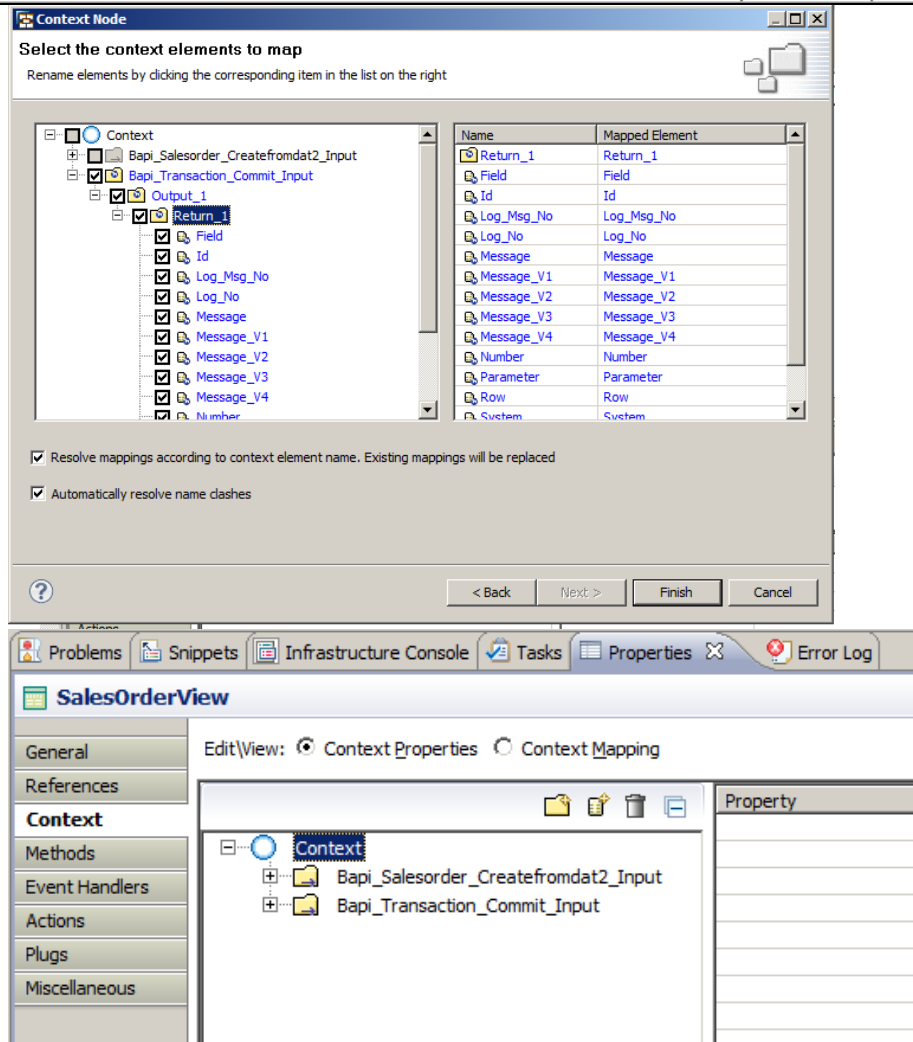
34.	Choose Model Binding. Click Next.																											
35.	Choose Bapi_Transaction_Commit_Input.																											
36.	Choose the appropriate node. Finish.	 <table><thead><tr><th>Name</th><th>Model Class</th></tr></thead><tbody><tr><td>Return_1</td><td>Bapiret2</td></tr><tr><td>Field</td><td>Field</td></tr><tr><td>Id</td><td>Id</td></tr><tr><td>Log_Msg_No</td><td>Log_Msg_No</td></tr><tr><td>Log_No</td><td>Log_No</td></tr><tr><td>Message</td><td>Message</td></tr><tr><td>Message_V1</td><td>Message_V1</td></tr><tr><td>Message_V2</td><td>Message_V2</td></tr><tr><td>Message_V3</td><td>Message_V3</td></tr><tr><td>Message_V4</td><td>Message_V4</td></tr><tr><td>Number</td><td>Number</td></tr><tr><td>Parameter</td><td>Parameter</td></tr></tbody></table>	Name	Model Class	Return_1	Bapiret2	Field	Field	Id	Id	Log_Msg_No	Log_Msg_No	Log_No	Log_No	Message	Message	Message_V1	Message_V1	Message_V2	Message_V2	Message_V3	Message_V3	Message_V4	Message_V4	Number	Number	Parameter	Parameter
Name	Model Class																											
Return_1	Bapiret2																											
Field	Field																											
Id	Id																											
Log_Msg_No	Log_Msg_No																											
Log_No	Log_No																											
Message	Message																											
Message_V1	Message_V1																											
Message_V2	Message_V2																											
Message_V3	Message_V3																											
Message_V4	Message_V4																											
Number	Number																											
Parameter	Parameter																											

## STEP 8. ADD CONTEXT USING MODEL BINDING IN SALES ORDER VIEW

37.	Open Context tab of Sales Order View. Right click on Context, <i>New &gt; Node</i> .	
38.	Choose Mapping. Click Next.	
39.	Choose <i>SOModelController</i> as required controller. Click OK.	

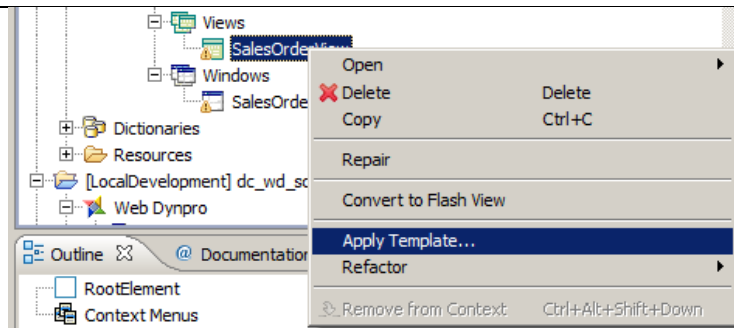
40.	Choose <b>Bapi_Salesorder_Creat efromdat2_Input</b> .	
41.	Select the context elements to map. Click Finish.	
42.	Repeat the process for <b>Bapi_Transaction_Com mit_Input</b> .	

43. Choose the node.  
As the result, context of SalesOrderView will contains two new nodes from context mapping to model.

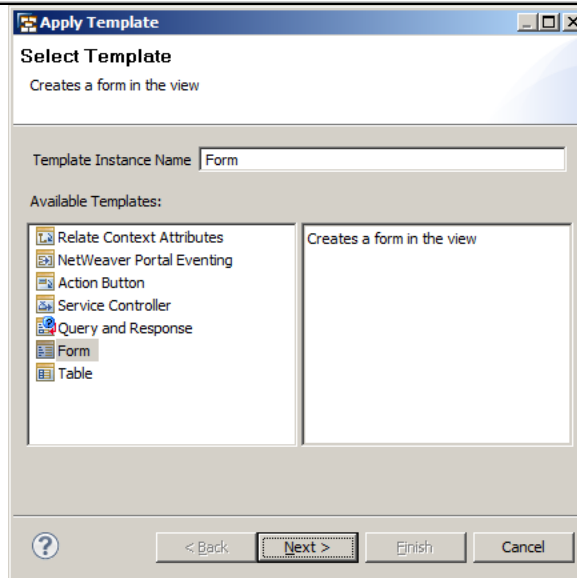


## STEP 9. CREATE UI WITH APPLY TEMPLATE

44. Right click on *SalesOrderView* > *Apply Template*.



45. Choose Form. Click Next.

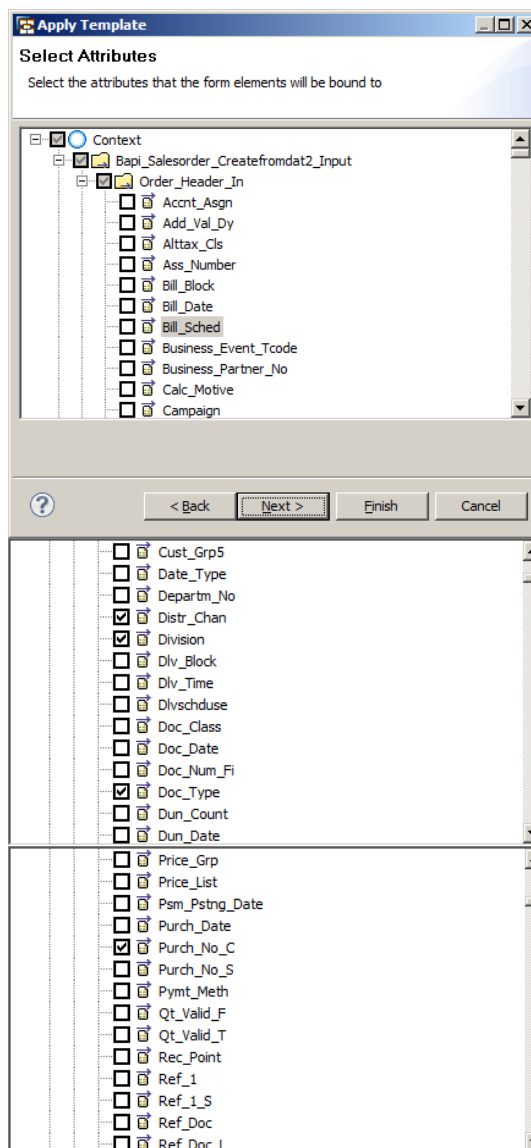


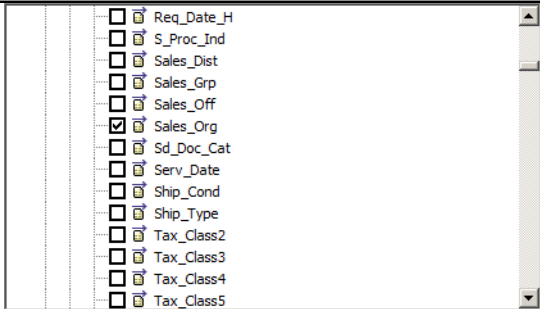
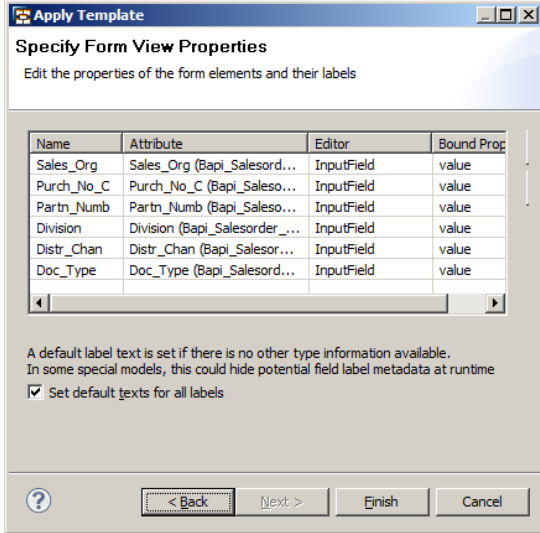
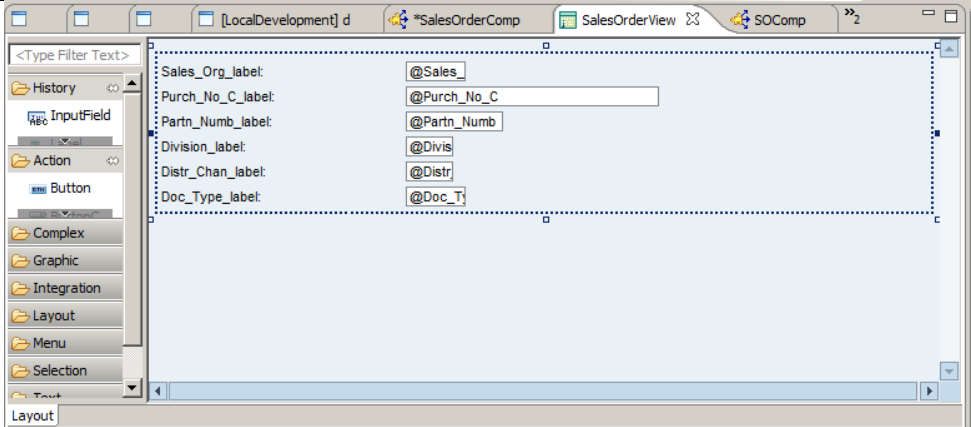
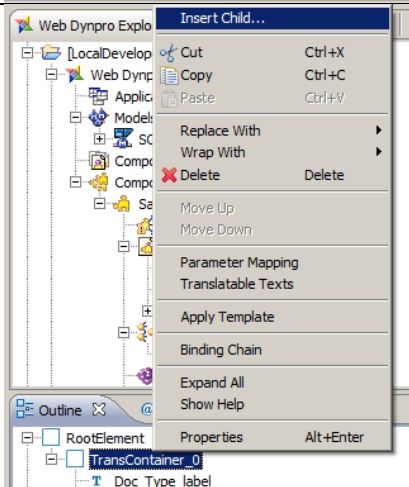
46. Here we will create UI for header data sales order:

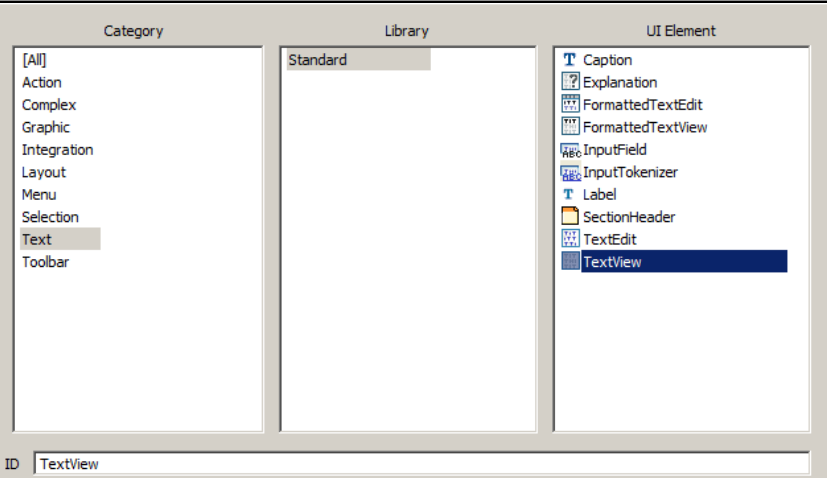
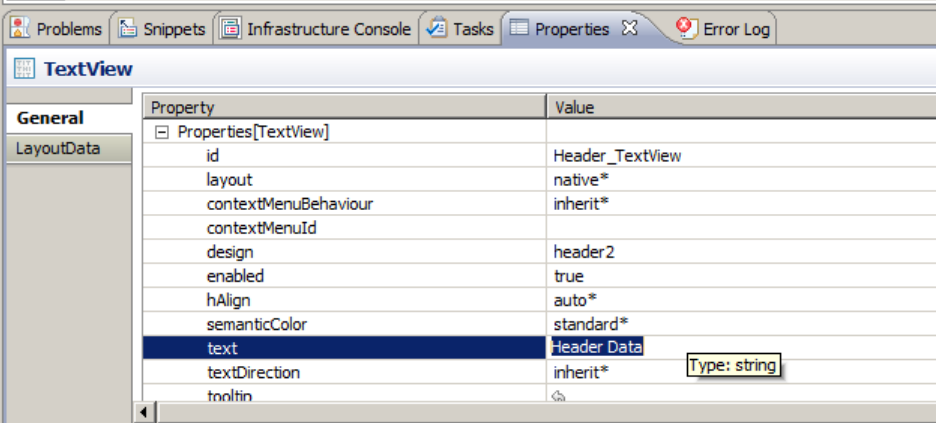
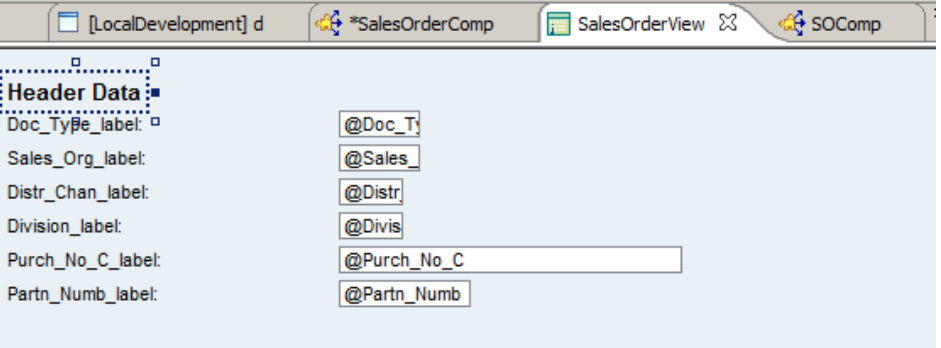
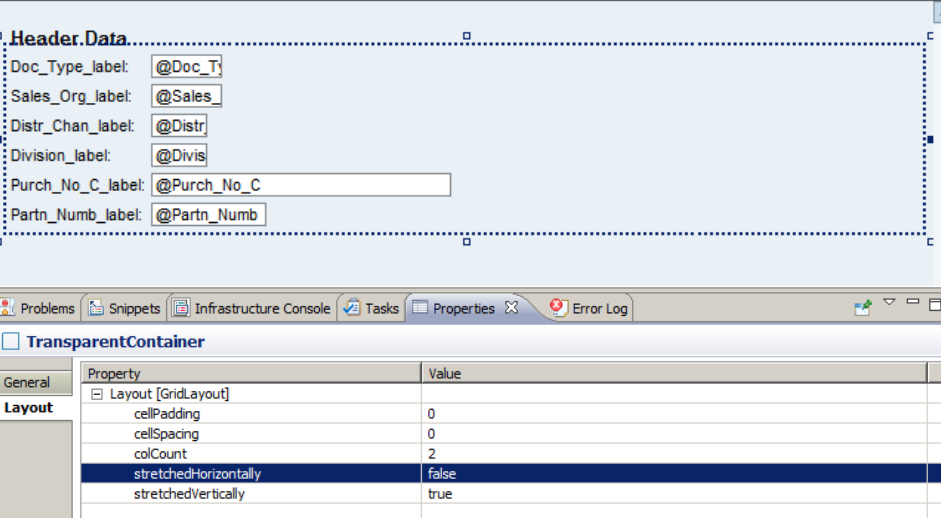
- Document Type
- Sales Organization
- Distribution Channel
- Division
- Purchase Order No.

Choose from Bapi\_Salesorder\_Createfromdat2\_Input > Order\_Header\_In. Thick on the appropriate element from the context appear.

Click Next.



																															
47.	Click Finish.	 <table border="1"> <thead> <tr> <th>Name</th><th>Attribute</th><th>Editor</th><th>Bound Prop</th></tr> </thead> <tbody> <tr> <td>Sales_Org</td><td>Sales_Org (Bapi_Salesord...</td><td>InputField</td><td>value</td></tr> <tr> <td>Purch_No_C</td><td>Purch_No_C (Bapi_Saleso...</td><td>InputField</td><td>value</td></tr> <tr> <td>Partn_Numb</td><td>Partn_Numb (Bapi_Saleso...</td><td>InputField</td><td>value</td></tr> <tr> <td>Division</td><td>Division (Bapi_Salesorder...</td><td>InputField</td><td>value</td></tr> <tr> <td>Distr_Chan</td><td>Distr_Chan (Bapi_Salesor...</td><td>InputField</td><td>value</td></tr> <tr> <td>Doc_Type</td><td>Doc_Type (Bapi_Salesord...</td><td>InputField</td><td>value</td></tr> </tbody> </table> <p><input checked="" type="checkbox"/> Set default texts for all labels</p>	Name	Attribute	Editor	Bound Prop	Sales_Org	Sales_Org (Bapi_Salesord...	InputField	value	Purch_No_C	Purch_No_C (Bapi_Saleso...	InputField	value	Partn_Numb	Partn_Numb (Bapi_Saleso...	InputField	value	Division	Division (Bapi_Salesorder...	InputField	value	Distr_Chan	Distr_Chan (Bapi_Salesor...	InputField	value	Doc_Type	Doc_Type (Bapi_Salesord...	InputField	value	
Name	Attribute	Editor	Bound Prop																												
Sales_Org	Sales_Org (Bapi_Salesord...	InputField	value																												
Purch_No_C	Purch_No_C (Bapi_Saleso...	InputField	value																												
Partn_Numb	Partn_Numb (Bapi_Saleso...	InputField	value																												
Division	Division (Bapi_Salesorder...	InputField	value																												
Distr_Chan	Distr_Chan (Bapi_Salesor...	InputField	value																												
Doc_Type	Doc_Type (Bapi_Salesord...	InputField	value																												
48.	Form with labels and text fields will automatically create based on context. The value of text fields also bound into context.																														
49.	We need little additional efforts to complete the UI. Here we will insert text field for title. On the Outline section of SalesOrderView right click on the header data's container > Insert Child.																														

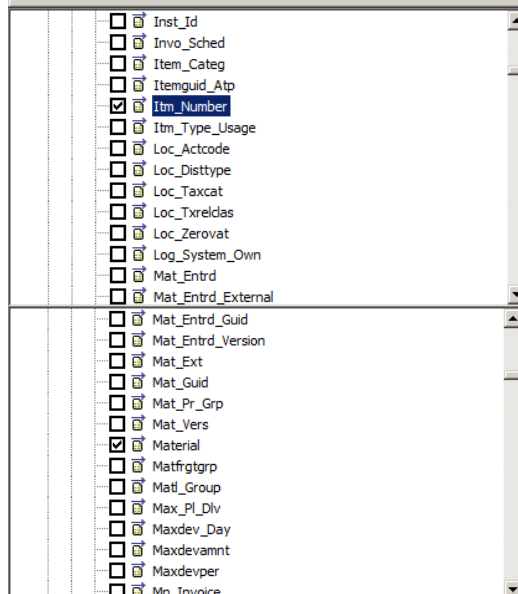
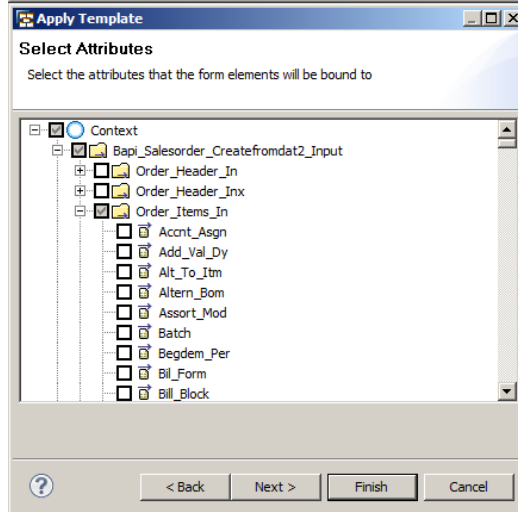
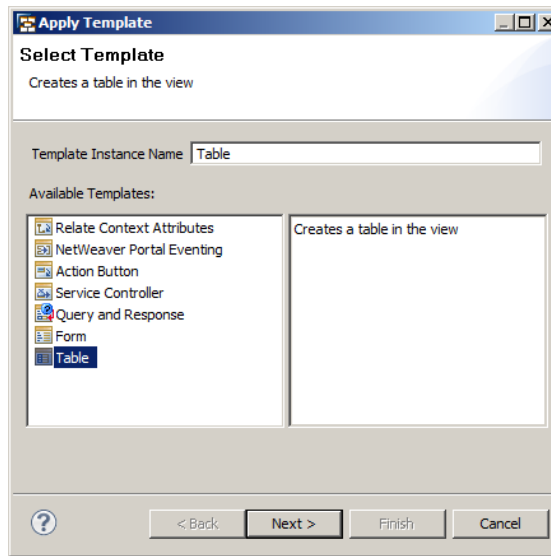
50.	Choose <i>Text</i> > <i>Text View</i> .	 <p>The screenshot shows the 'UI Element' library in an IDE. The 'Category' pane on the left has 'Text' selected. The 'Library' pane in the middle shows 'Standard'. The 'UI Element' pane on the right lists various elements, with 'TextView' highlighted at the bottom.</p>
51.	A new child will be generated. Rename the id property with "Header_TextView" and text property to "Header Data".	 <p>The screenshot shows the 'Properties' window for a 'TextView' element. The 'General' tab is active, and the 'Properties[TextView]' section is expanded. The 'id' property is set to 'Header_TextView', and the 'text' property is set to 'Header Data'. The 'text' property is highlighted, and a tooltip shows 'Type: string'.</p>
52.	The result will show like this.	 <p>The screenshot shows a visual representation of the 'Header Data' view. It displays several labels and their corresponding values: 'Doc_Type_label' with '@Doc_T', 'Sales_Org_label' with '@Sales', 'Distr_Chan_label' with '@Distr', 'Division_label' with '@Divis', 'Purch_No_C_label' with '@Purch_No_C', and 'Partn_Numb_label' with '@Partn_Numb'.</p>
53.	We can make change to the attribute of the property. For the example I change the Layout property to not to be able to stretch horizontally. Thus the property is set to "false".	 <p>The screenshot shows the 'Properties' window for a 'TransparentContainer' element. The 'Layout' tab is active, and the 'Layout [GridLayout]' section is expanded. The 'stretchedHorizontally' property is set to 'false', and the 'stretchedVertically' property is set to 'true'.</p>

54. Continue to create another UI, table for store item data.  
Now we choose Order\_Items\_In to be applied into template.

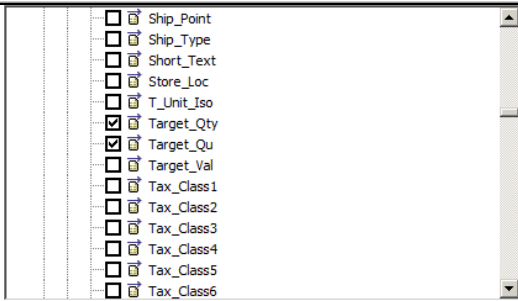
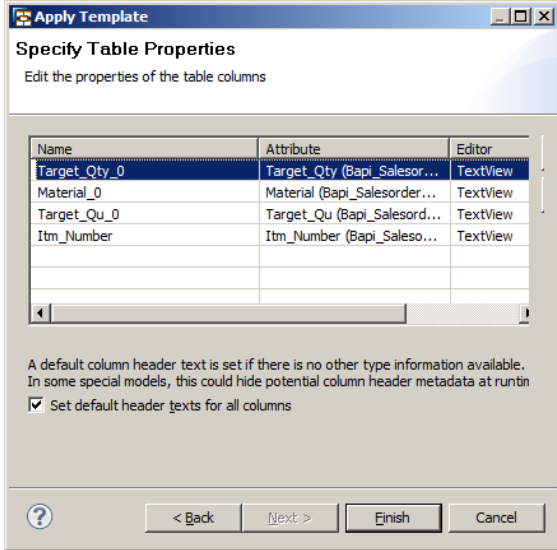
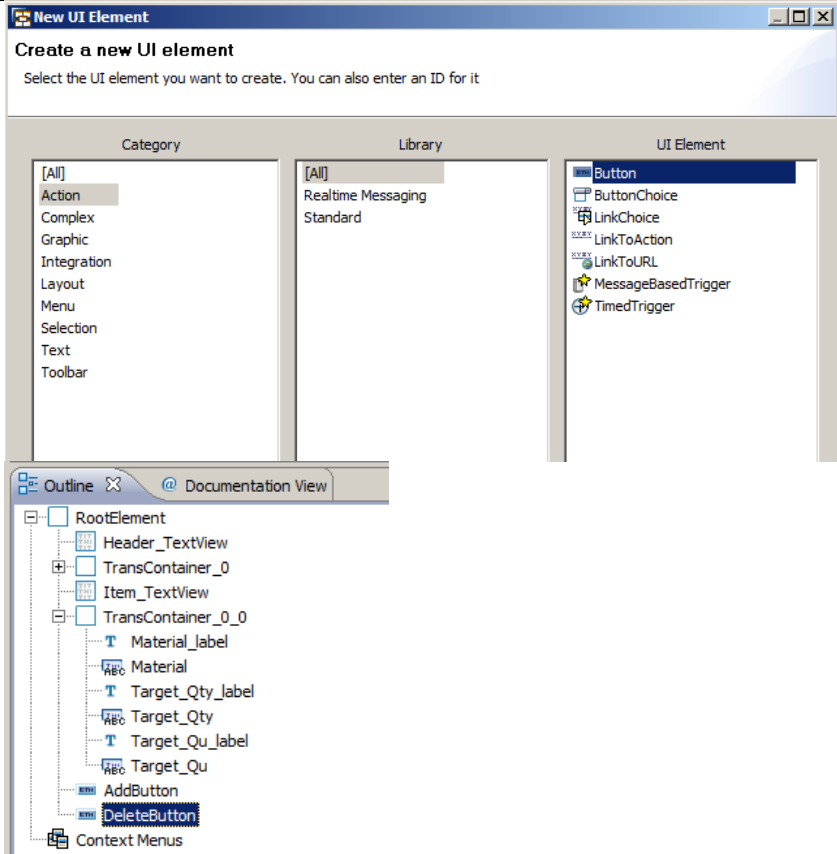
Right click on *SalesOrderView* > *Apply Template*.

Choose Table. Click Next.

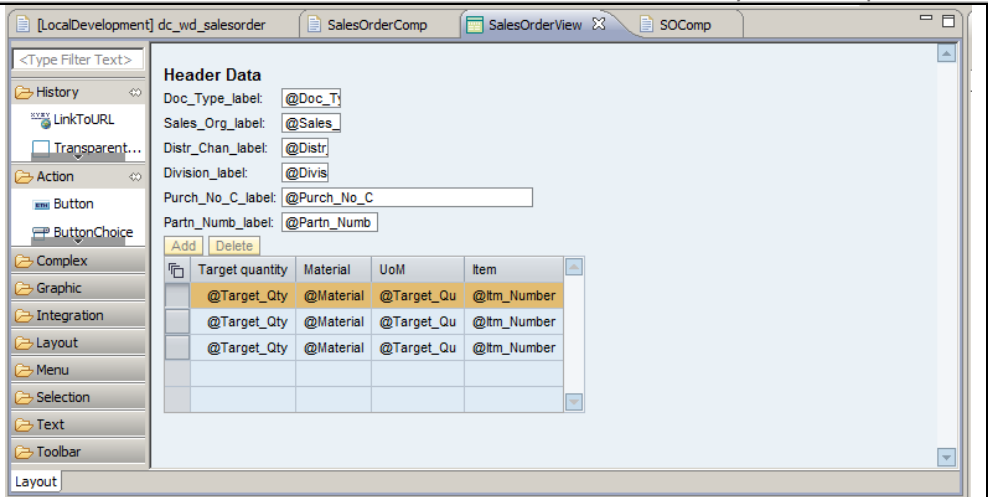
Choose Order\_Items\_In:  
 - Itm\_Number  
 - Material  
 - Target\_Qty  
 - Target\_Qu  
 Click Finish.





		
55.	Click Finish.	
56.	<p>Continue with create two additional button: "Add" and "Delete". On Outline, right click on item's container &gt; <i>Insert New Child</i>.</p> <p>Choose <i>Action &gt; Button</i>.</p> <p>Rename the default name identifier with "Add". Then copy this button to be pasted and renamed it to "Delete".</p>	

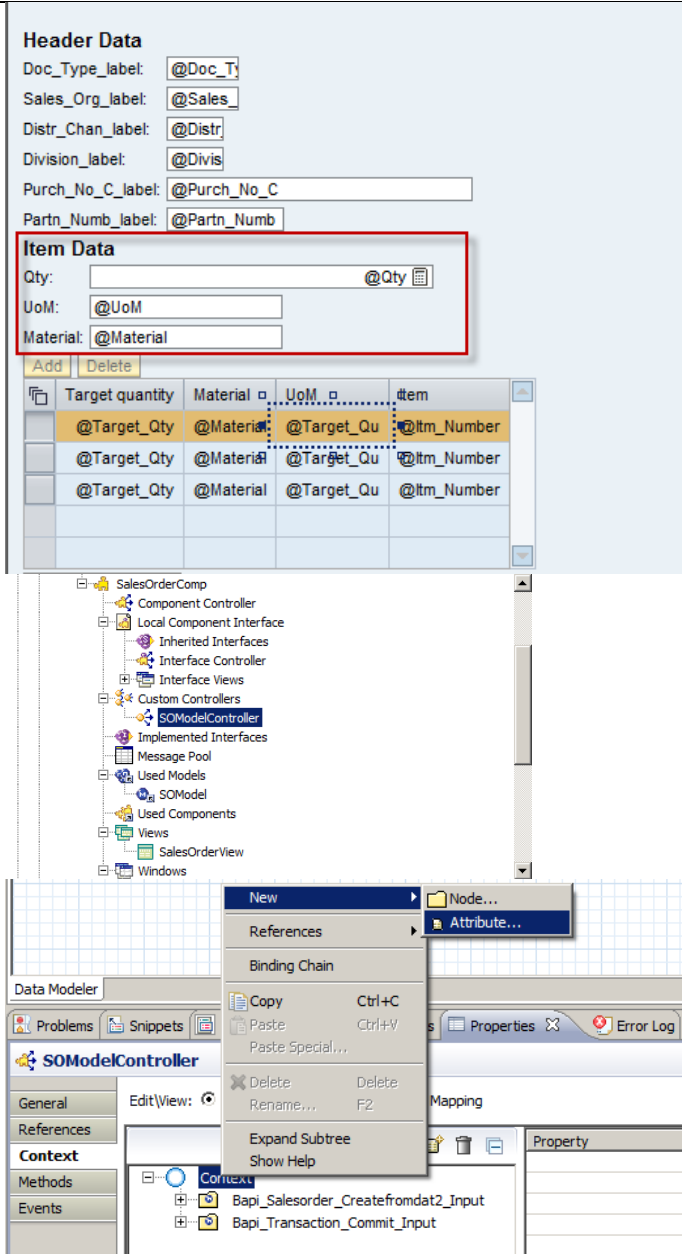
57. Arrange the order of the layout as appropriate thus the result will come like screenshot beside.

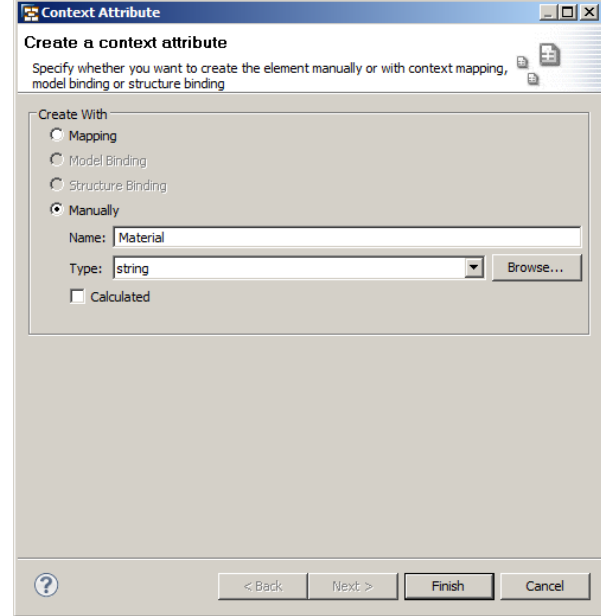
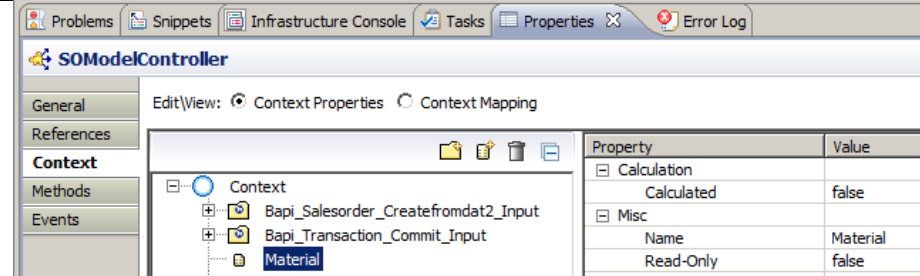
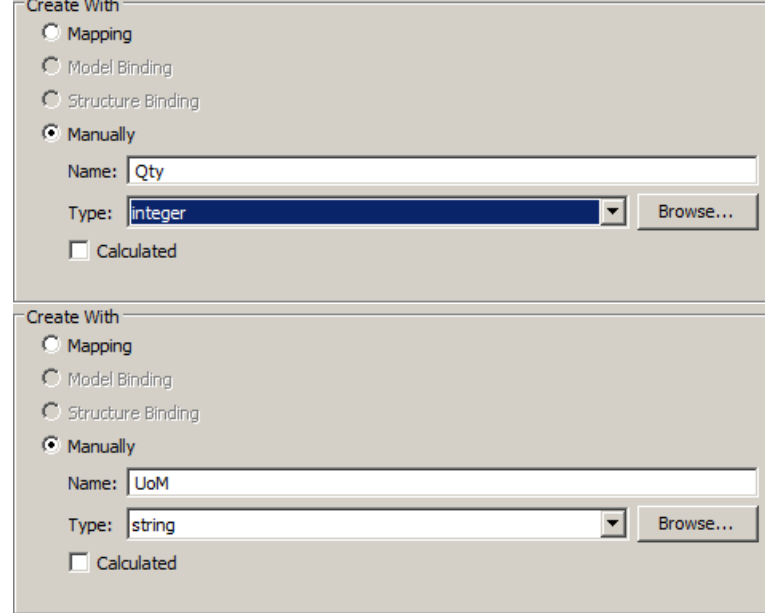


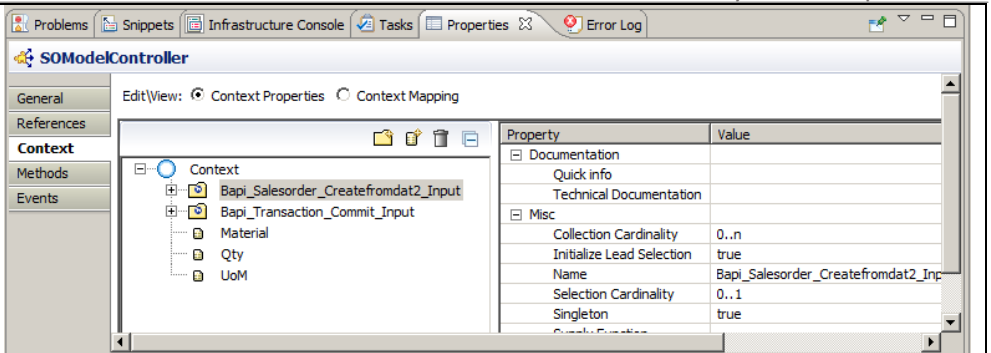
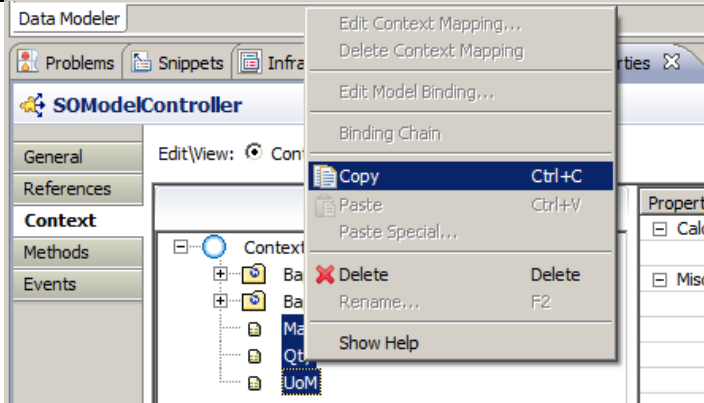
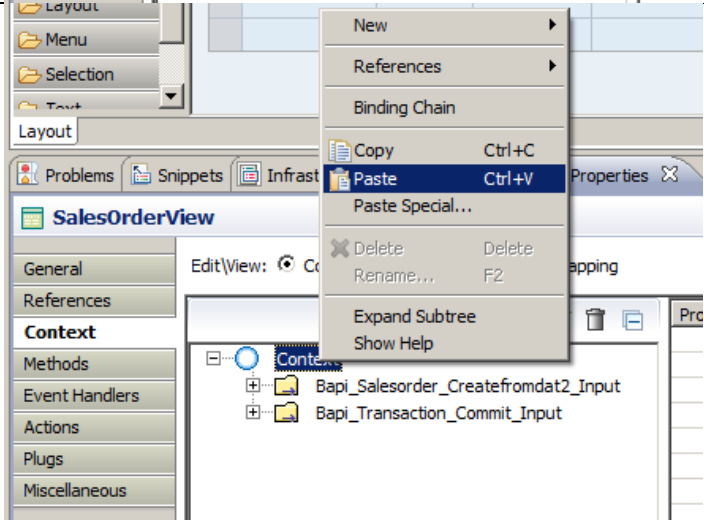
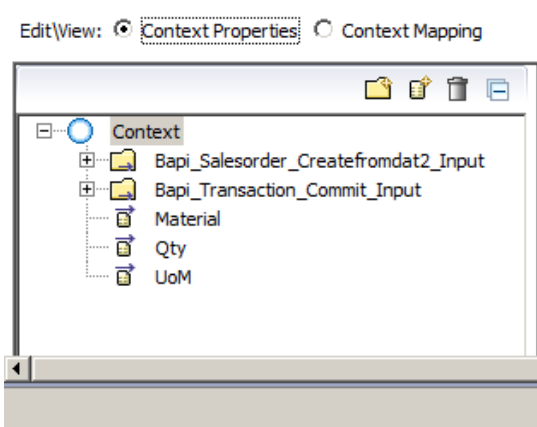
58. We need to create multiple items in item data that will display in table first before bound into context and send to BAPI.

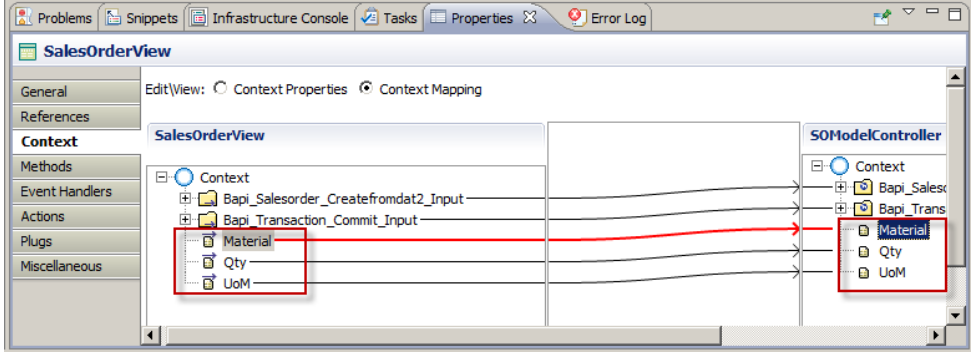
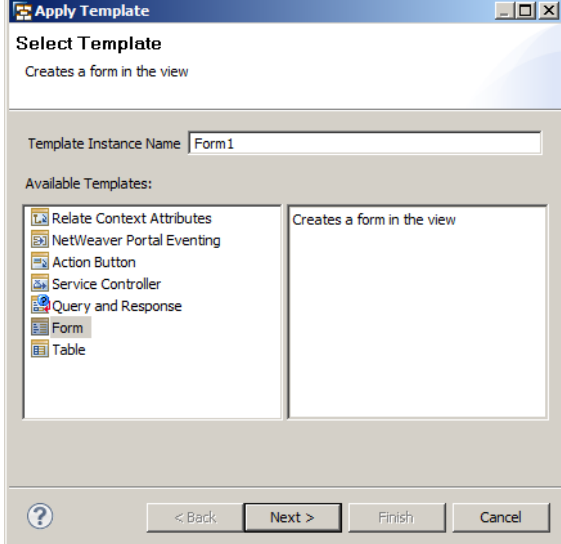
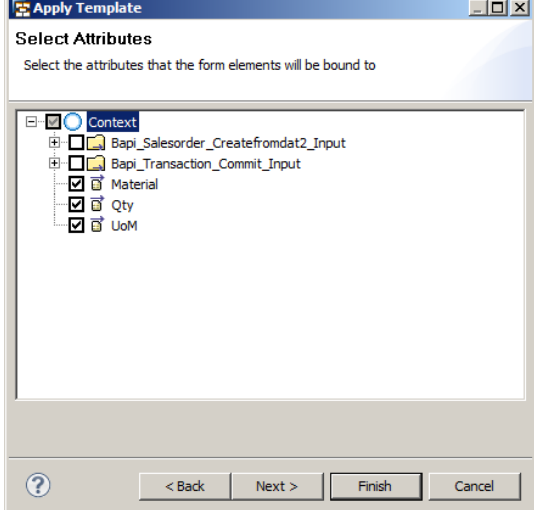
Thus, we need to create additional contexts to temporary stores the data before send into BAPI.

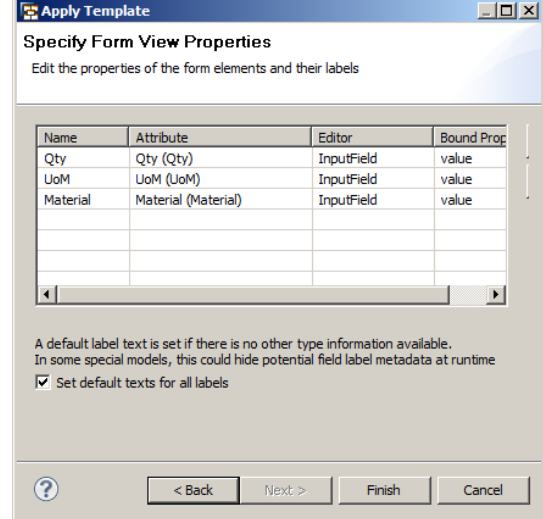
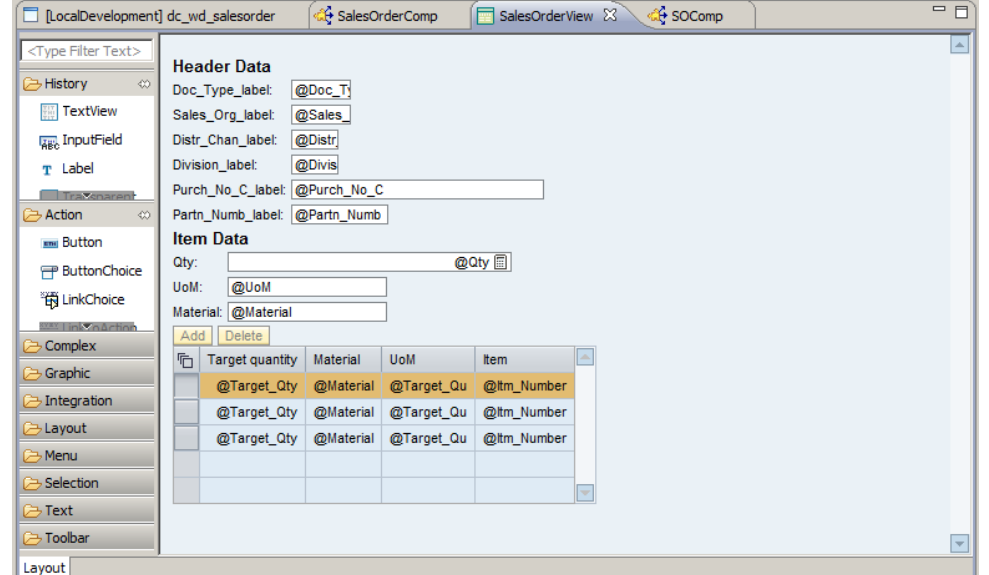
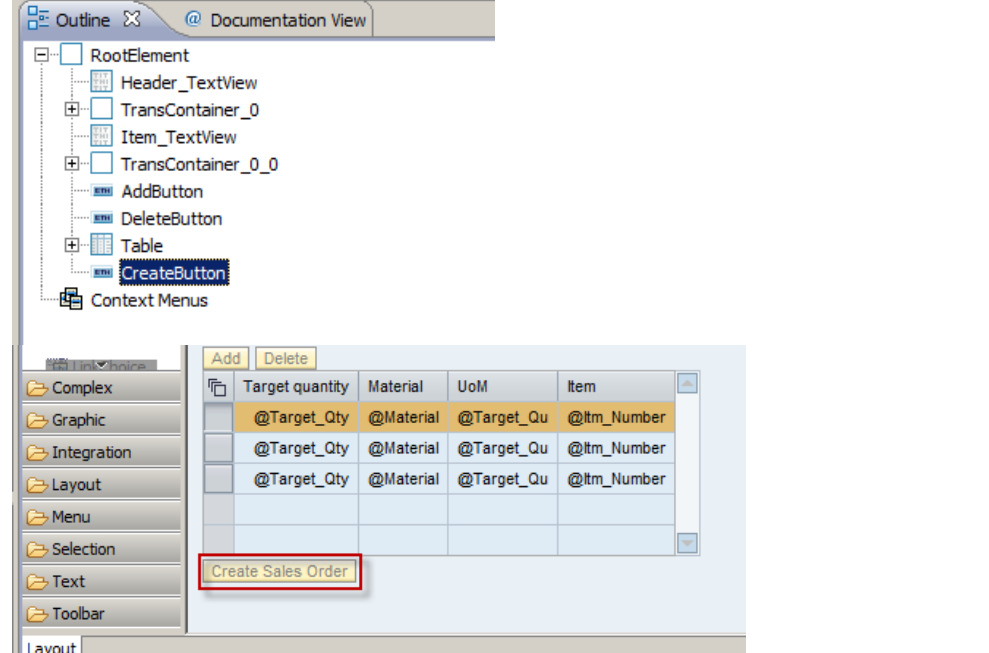
Double click on SOModelController. On Context tab, right click on *Context* > *New* > *Attribute*.

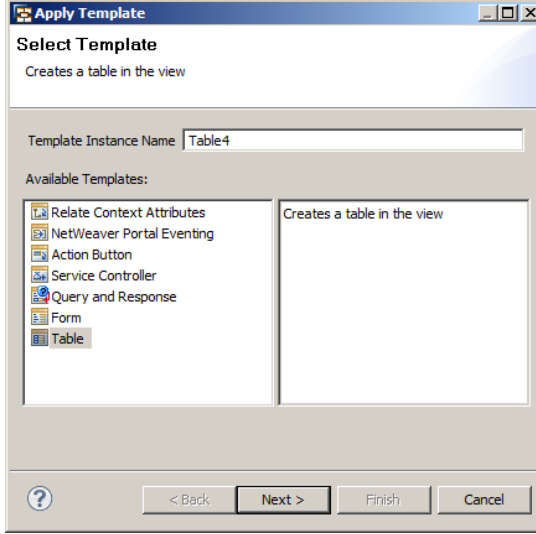
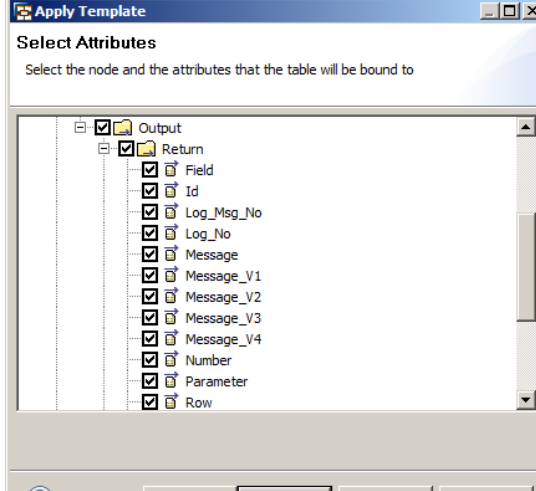
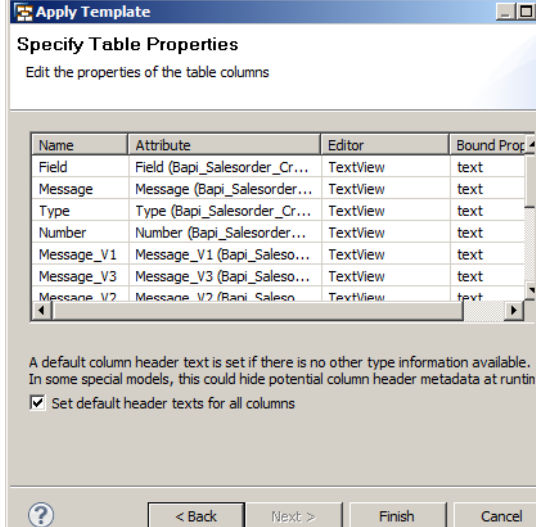


59.	<p>Choose Manually.</p> <p>Enter:</p> <p>Name = Material.</p> <p>Type = string.</p> <p>Click Finish.</p>														
60.	<p>New attribute will appear in context.</p>	 <table data-bbox="1115 917 1453 1060"><tr><th>Property</th><th>Value</th></tr><tr><td>Calculation</td><td></td></tr><tr><td>Calculated</td><td>false</td></tr><tr><td>Misc</td><td></td></tr><tr><td>Name</td><td>Material</td></tr><tr><td>Read-Only</td><td>false</td></tr></table>	Property	Value	Calculation		Calculated	false	Misc		Name	Material	Read-Only	false	
Property	Value														
Calculation															
Calculated	false														
Misc															
Name	Material														
Read-Only	false														
61.	<p>Repeat the process to create manually attribute to create two another attributes for quantity and UoM.</p> <p>Attributes:</p> <p>Name = Qty</p> <p>Type = integer</p> <p>Name = UoM</p> <p>Type = string</p>														

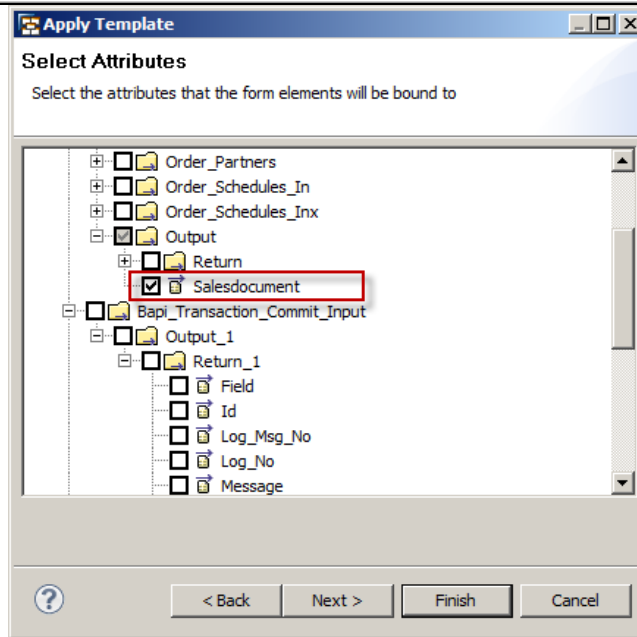
		
62.	Choose the three attributes by left click on each attribute by pressing shift keyboard. Then right click > Copy.	
63.	Open the properties of SalesOrderView, and paste this three attributes into context.	 <p>Thus, the context will have three new attributes copy from custom controller.</p> 

<p>64. Create context mapping.</p> <p>Open Context tab, choose Context Mapping.</p> <p>Drag each the new attribute from SalesOrderView to SOModelController to create context mapping.</p>	
<p>65. After finished mapping the context to custom controller, then you can create another UI for item data.</p> <p>Right click on <i>SalesOrder View</i> &gt; <i>Apply Template</i>.</p> <p>Choose Form. Click Next.</p>	
<p>66. Choose Material, Qty, and UoM. Click Next.</p>	

67.	Click Finish.	
68.	Do some order of the arrangement of UIs thus the final result will show like this.	
69.	<p>To trigger action to call BAPI_SALES_CREATEFROMDAT2 we need a button that will generate event.</p> <p>Create a button and rename it to “Create Sales Order”. Position the button below the Item Data table.</p>	

70.	<p>Last, create a table for display log of creation of sales order.</p> <p>Right click on SalesOrderView &gt; Apply Template.</p> <p>Click Next.</p>	
71.	<p>Choose from node Bapi_Salesorder_Creatfromdat2 &gt; Output &gt; Return.</p> <p>Click Next.</p>	
72.	<p>Click Finish.</p> <p>SAVE.</p>	

73. Last, add a label and an input field to display *Sales Document Number*. Use apply template and bound from node Output > Salesdocument.



The final result of our layout.

der SalesOrderComp SalesOrderView SOComp

**Header Data**

Doc\_Type\_label: @Doc\_Ty

Sales\_Org\_label: @Sales\_

Distr\_Chanel\_label: @Distr

Division\_label: @Divis

Purch\_No\_C\_label: @Purch\_No\_C

Partn\_Numb\_label: @Partn\_Numb

**Item Data**

Qty: @Qty

UoM: @UoM

Material: @Material

Add Delete

Target quantity	Material	UoM	Item
@Target_Qty	@Material	@Target_Qu	@itm_Number
@Target_Qty	@Material	@Target_Qu	@itm_Number
@Target_Qty	@Material	@Target_Qu	@itm_Number

Create Sales Order

Fid.	Message	MsgType	Msg.No.	Message Variable	Message Variable	Message Variable	Msg.no	Log number	Message Variable
@Field	@Message	@Type	@Number	@Message_V1	@Message_V3	@Message_V2	@Log_Msg_No	@Log_No	@Message_V4
@Field	@Message	@Type	@Number	@Message_V1	@Message_V3	@Message_V2	@Log_Msg_No	@Log_No	@Message_V4
@Field	@Message	@Type	@Number	@Message_V1	@Message_V3	@Message_V2	@Log_Msg_No	@Log_No	@Message_V4

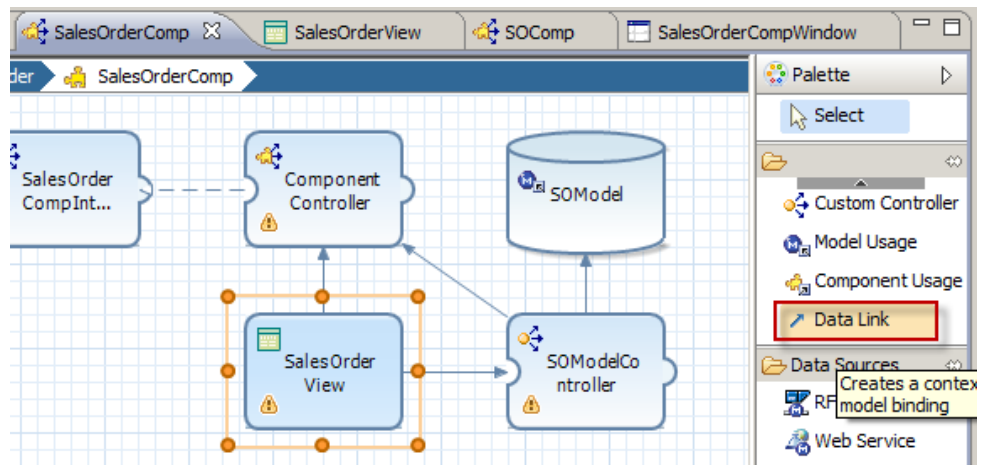
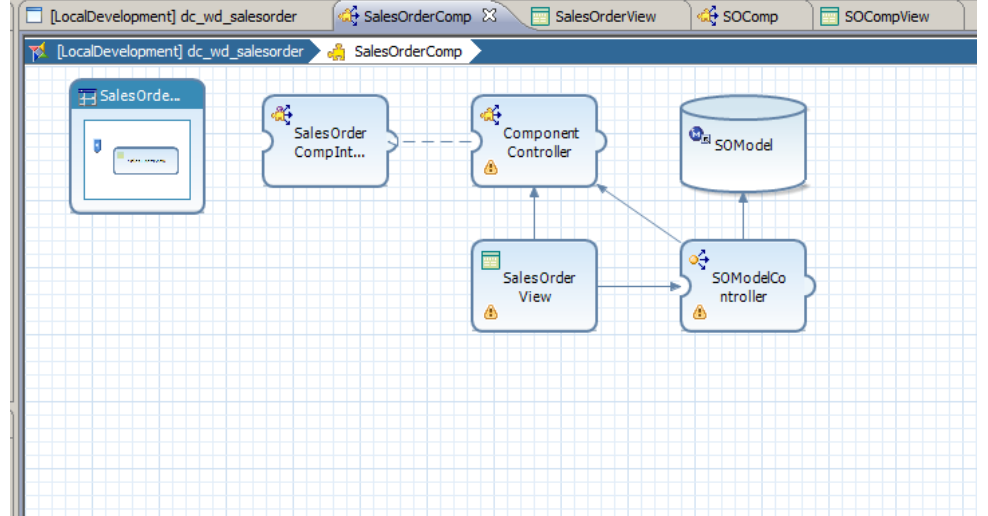
Salesdocument\_label: @Salesdocume



## STEP 10. CREATE LINK TO COMPONENT CONTROLLER

74. In this step we will create link of our custom controller and view to component. Thus SOModelController will have data link to Component Controller and SalesOrderView too.

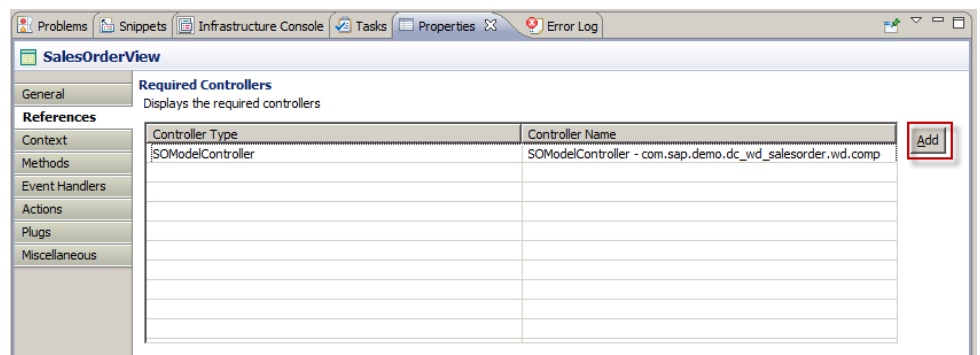
The first way is by drag and drop data link. Click on SalesOrderView, then from Palette appear choose **Data Link**. If cursor already change Click SalesOrderView to ComponentController.



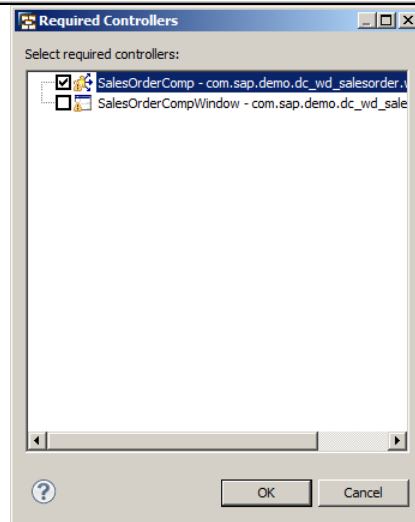
Or

The another method by accessing through *References* in SalesOrderView's properties.

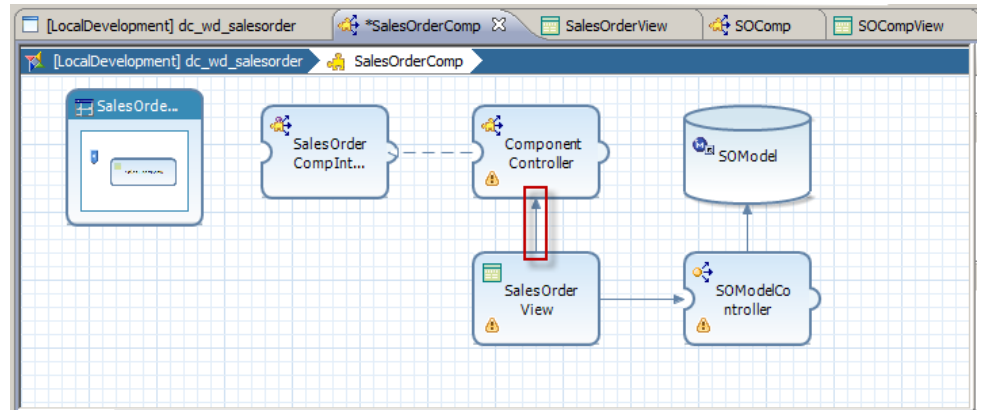
Click Add button.



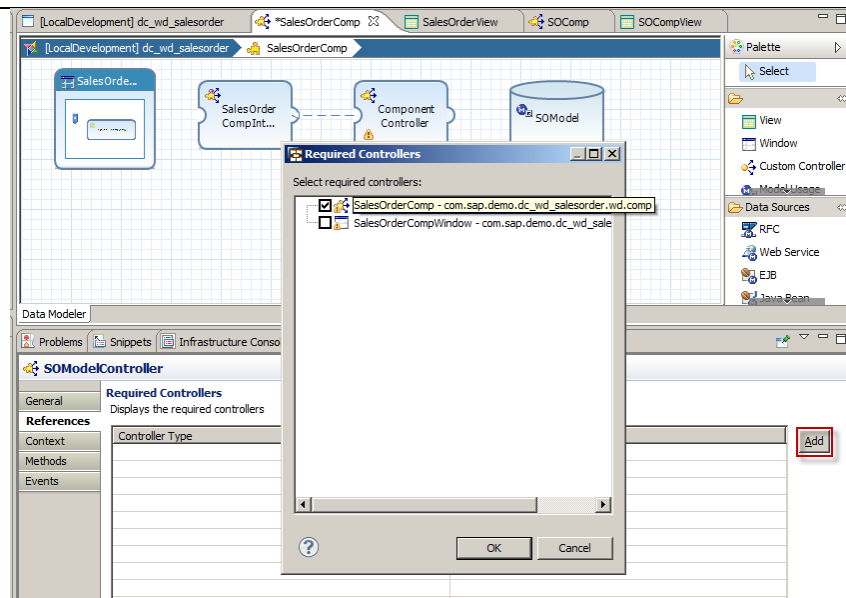
On pop up window appear, choose SalesOrderComp. Click OK.

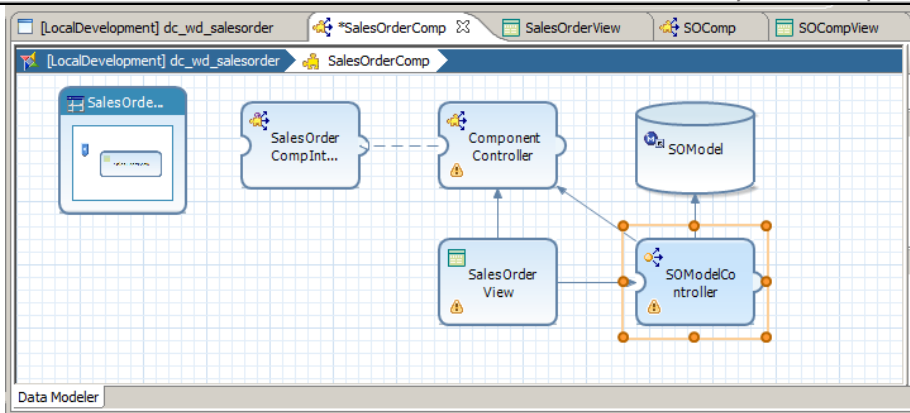


Result:



75. Repeat the same way to SOModelController.  
SAVE.

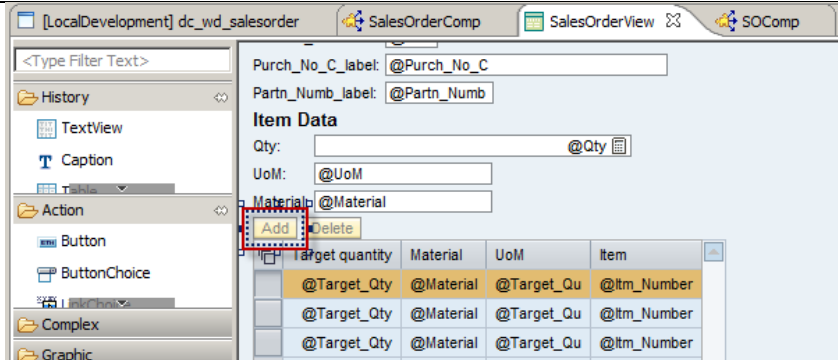




## STEP 11. ADDING EVENT TO BUTTONS

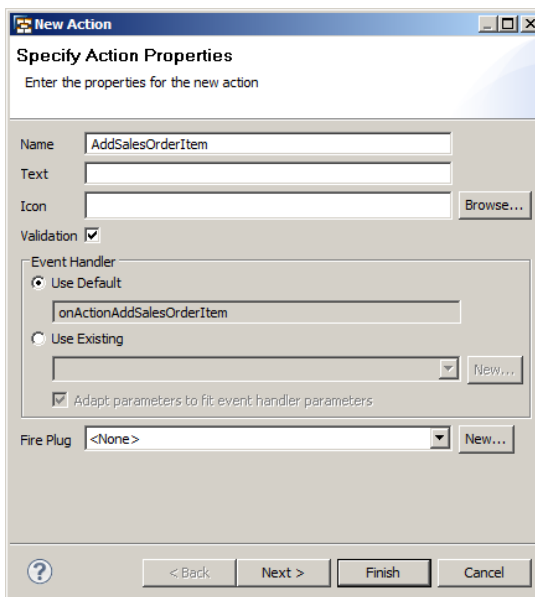
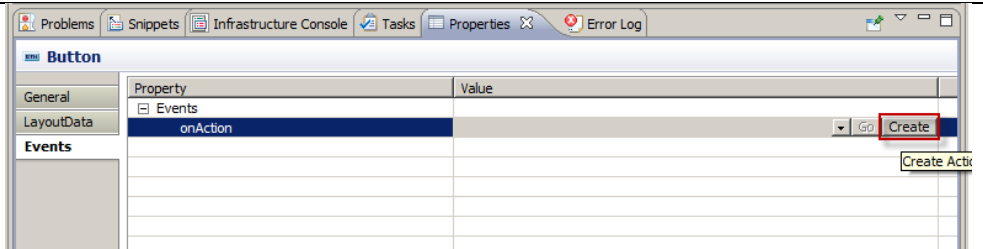
76. From the step before we only created layout and the buttons still not be able to work. Here we will add some java code to make these button do their job.

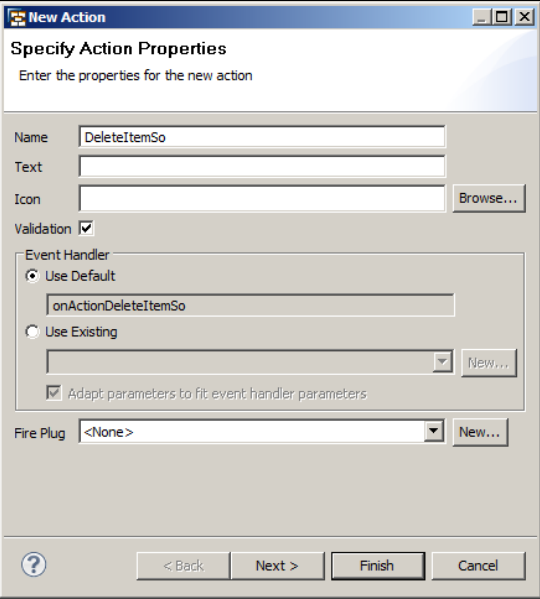
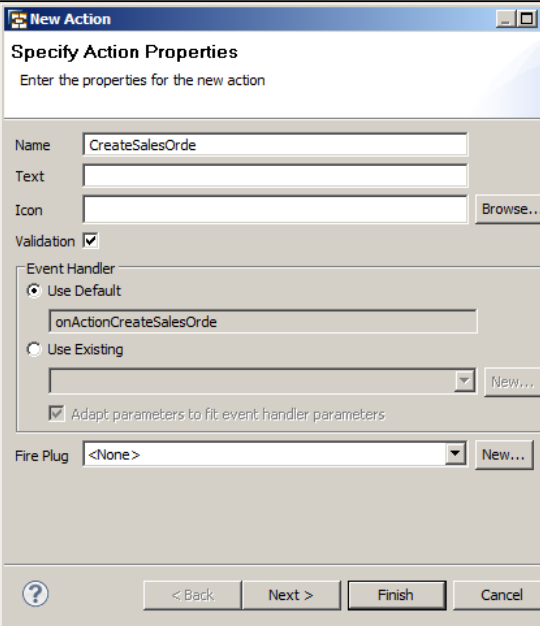
First, open the layout we have, click on "Add" button.



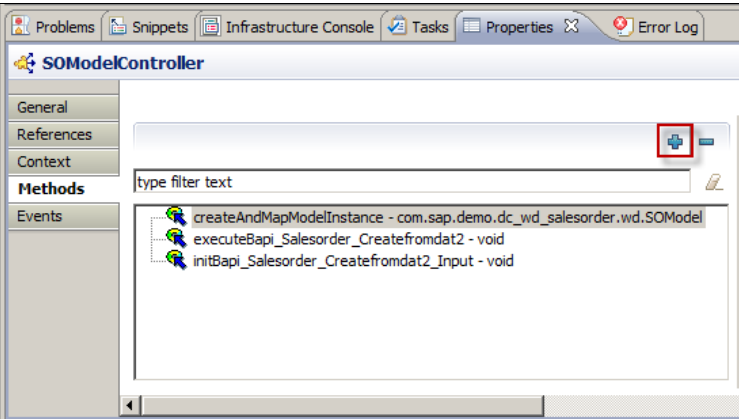
77. After properties window of Add button appeared, click on Events tab and create new action.

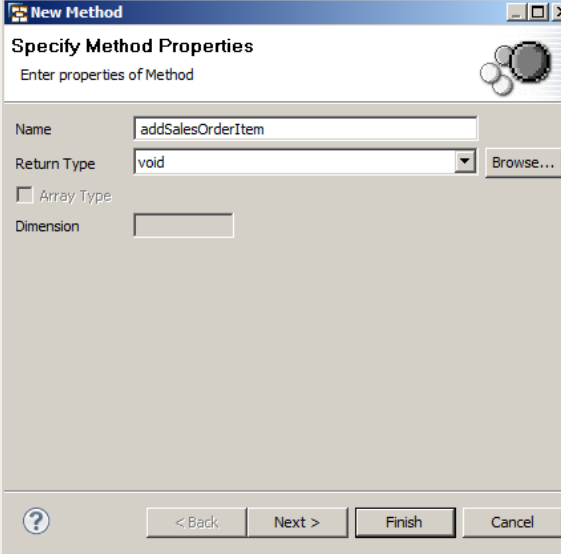
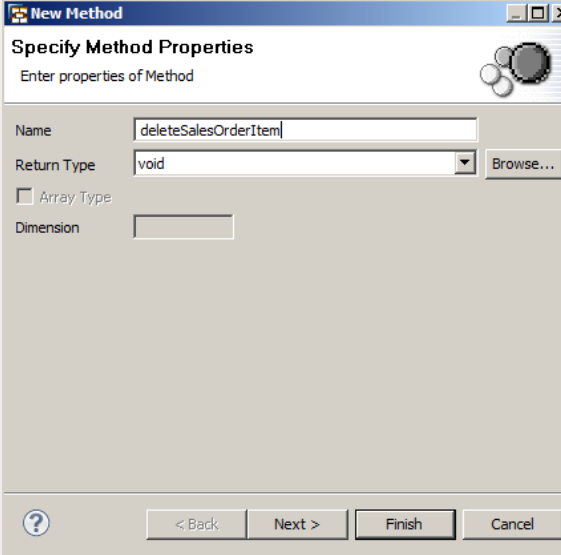
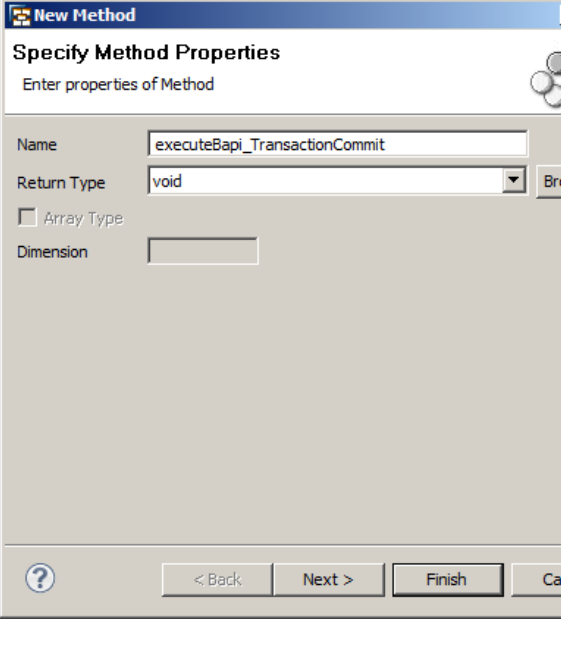
Type "AddSalesOrderItem" on field Name. It automatically create method that will we use later. Click Finish.

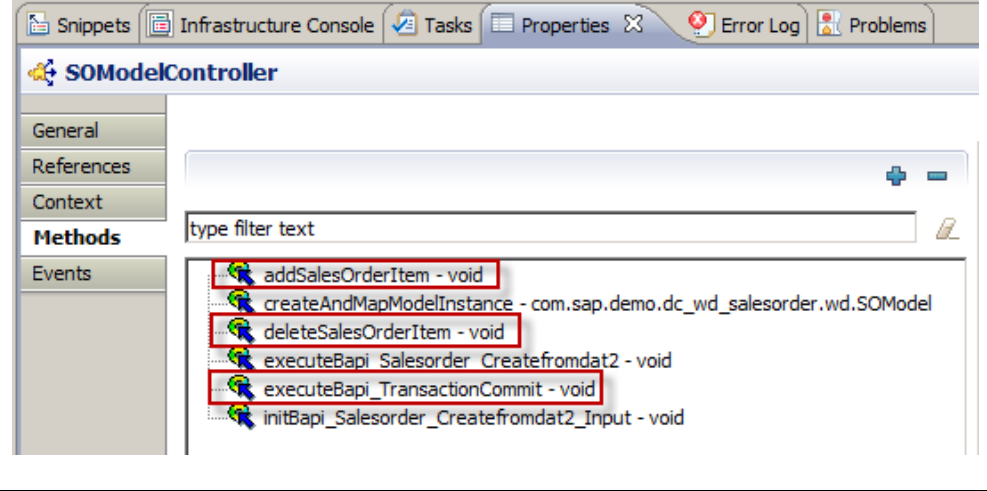
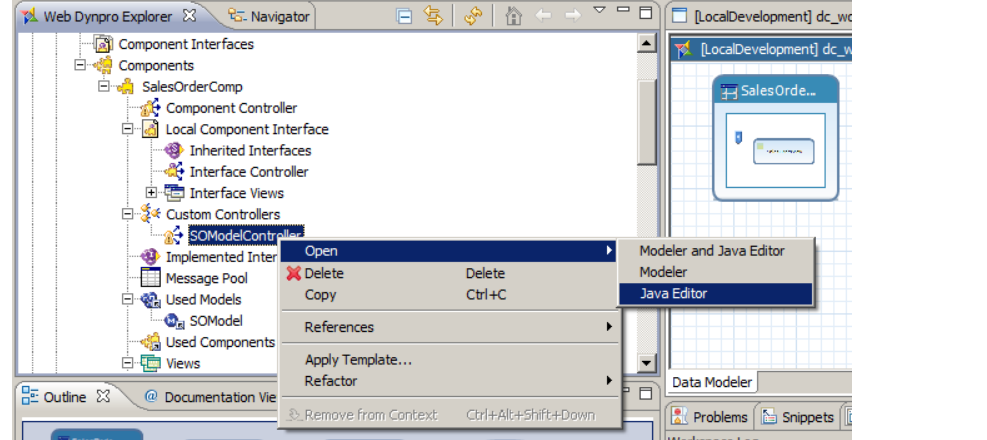


78.	Continue the same process for “Delete” button.	 <p>The screenshot shows the 'New Action' dialog box titled 'Specify Action Properties'. The 'Name' field is 'DeleteItemSo'. The 'Text' field is empty. The 'Icon' field is empty with a 'Browse...' button. The 'Validation' checkbox is checked. Under 'Event Handler', 'Use Default' is selected, and the text 'onActionDeleteItemSo' is entered. The 'Use Existing' option is unselected. The 'Adapt parameters to fit event handler parameters' checkbox is checked. The 'Fire Plug' dropdown is set to '&lt;None&gt;'. At the bottom are buttons for '&lt; Back', 'Next &gt;', 'Finish', and 'Cancel'.</p>	
79.	Last, for “Create Sales Order” button too.	 <p>The screenshot shows the 'New Action' dialog box titled 'Specify Action Properties'. The 'Name' field is 'CreateSalesOrde'. The 'Text' field is empty. The 'Icon' field is empty with a 'Browse...' button. The 'Validation' checkbox is checked. Under 'Event Handler', 'Use Default' is selected, and the text 'onActionCreateSalesOrde' is entered. The 'Use Existing' option is unselected. The 'Adapt parameters to fit event handler parameters' checkbox is checked. The 'Fire Plug' dropdown is set to '&lt;None&gt;'. At the bottom are buttons for '?', '&lt; Back', 'Next &gt;', 'Finish', and 'Cancel'.</p>	

## STEP 11. CREATE METHOD IN CUSTOM CONTROLLER

80.	Open custom controller, on Methods tab, click on plus sign. Here we will add method that will be used to add java code.	 <p>The screenshot shows the 'SOMModelController' IDE. The 'Methods' tab is selected in the left sidebar. The main area shows a list of methods: 'createAndMapModelInstance - com.sap.demo.dc_wd_salesorder.wd.SOMModel', 'executeBapi_Salesorder_Createfromdat2 - void', and 'initBapi_Salesorder_Createfromdat2_Input - void'. A red box highlights a plus sign icon in the top right corner of the methods list area.</p>
-----	-------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

81.	Type "addSalesOrderItem". This method will we use to write some logic to "Add" button. Click Finish.		
82.	Add another method "deleteSalesOrderItem".		
83	Since we not yet add method for Bapi_Transaction_Commit. Now, add a method in custom controller for this BAPI. Click Finish.		

84.	<p>For the three result, we now have three new methods in our custom controller. We will add some logic of this method later.</p>	
<h2>STEP 12. IMPLEMENT JAVA CODE IN CUSTOM CONTROLLER</h2>		
85.	<p>After we add methods that we need to write some logic to be applied on our buttons we created before, now is time to implement the logic.</p> <p>Right click on <i>SOModelController</i> &gt; <i>Open</i> &gt; <i>Java Editor</i>.</p>	
86.	<p>After java code open, find method <b>wdDoInit()</b> and add some java code.</p> <p>In this method we will make an initialization when the first time java program is called and run.</p>	<pre> public void wdDoInit() {     //@@begin wdDoInit()     //\$begin Service Controller(683563848)     initBapi_Salesorder_Createfromdat2_Input();      SOModel soModel = new SOModel();     Bapi_Salesorder_Createfromdat2_Input input = new Bapi_Salesorder_Createfromdat2_Input(soModel);     wdContext.nodeBapi_Salesorder_Createfromdat2_Input().bind(input);      Bapisdhd1 header = new Bapisdhd1(soModel);     Bapisdhd1X headerx = new Bapisdhd1X(soModel);     Bapisditm item = new Bapisditm(soModel);     Bapisditmx itemx = new Bapisditmx(soModel);     Bapiiparnr partner = new Bapiiparnr(soModel);     Bapischdl schedule = new Bapischdl(soModel);     Bapischdlx schedulx = new Bapischdlx(soModel);     partner.setPartn_Role("AG");      input.setOrder_Header_In(header);     input.setOrder_Header_Inx(headerx);      input.addOrder_Partners(partner);      Bapi_Transaction_Commit_Input inputCommit = new Bapi_Transaction_Commit_Input(soModel);     inputCommit.setWait(true);     wdContext.nodeBapi_Transaction_Commit_Input().bind(inputCommit);      //\$end     //@@end } </pre>

```

public void wdDoInit()
{
    //@@begin wdDoInit()
    //$begin Service Controller(683563848)
    initBapi_Salesorder_Createfromdat2_Input();

    SOModel soModel = new SOModel();
    Bapi_Salesorder_Createfromdat2_Input input = new
    Bapi_Salesorder_Createfromdat2_Input(soModel);

    wdContext.nodeBapi_Salesorder_Createfromdat2_Input().bind
    d(input);

    Bapisdhd1 header = new Bapisdhd1(soModel);
    Bapisdhd1X headerx = new Bapisdhd1X(soModel);
    Bapisditm item = new Bapisditm(soModel);
    Bapisditmx itemx = new Bapisditmx(soModel);
    Bapiparnr partner = new Bapiparnr(soModel);
    Bapischdl schedule = new Bapischdl(soModel);
    Bapischdlx schedulex = new Bapischdlx(soModel);

    partner.setPartn_Role("AG");

    input.setOrder_Header_In(header);
    input.setOrder_Header_Inx(headerx);

    input.addOrder_Partners(partner);

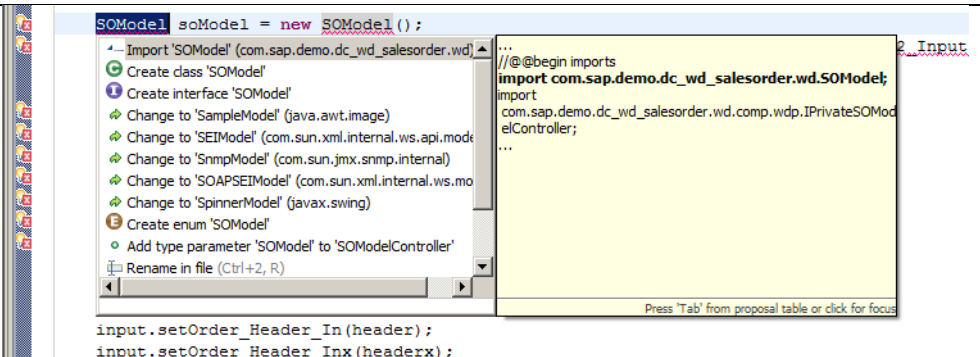
    Bapi_Transaction_Commit_Input inputCommit = new
    Bapi_Transaction_Commit_Input(soModel);
    inputCommit.setWait(true);

    wdContext.nodeBapi_Transaction_Commit_Input().bind
    (inputCommit);

    //$end
    //@@end
}

```

87. If there is still appear error, click on bubble lamp that appear beside error and do appropriate import.



88. Finally SAVE.

89. Now we will add logic to the methods we created before.

Double click on custom controller. After properties appear, open the Methods tab.

First right click on method `executeBapi_Salesorder_Createfromdat2()`, *Navigate To > Implementation.*

SAVE.

The screenshot shows the SAP Web Dynpro IDE interface. On the left, the 'Methods' tab is selected, displaying a list of methods. The method `executeBapi_Salesorder_Createfromdat2 - void` is highlighted. A right-click context menu is open over this method, with the 'Navigate To' option selected, leading to the 'Implementation' view (F3).

The 'Implementation' view shows the following code:

```

public void executeBapi_Salesorder_Createfromdat2( ) {
    //@@begin executeBapi_Salesorder_Createfromdat2()
    //$begin Service Controller(-460758325)

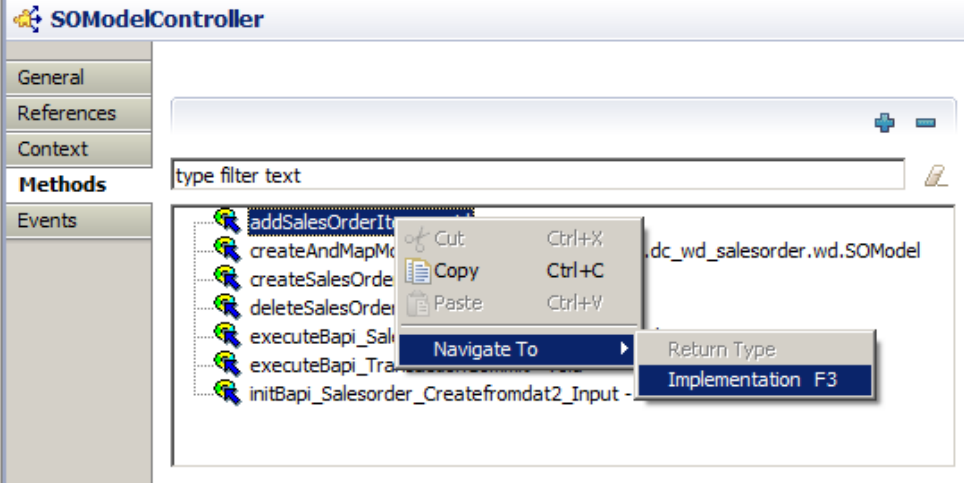
    IWDMessageManager manager = wdComponentAPI.getMessageManager ();

    try
    {
        wdContext.currentBapi_Salesorder_Createfromdat2_InputElement().modelObject().execute();
        wdContext.nodeOutput().invalidate();
    } catch (ARFC2ModelExecuteException e) {
        wdComponentAPI.getMessageManager().reportException ( e.getMessage());
    }
    //$end
    //@@end
}

```

The code is displayed in a light blue box, indicating it is the selected implementation.



90.	<p>Find implementation of method <b>executeBapi_TransactionCommit()</b> and add some additional java code.</p> <p>SAVE.</p>	<pre> public void executeBapi_TransactionCommit( ) {     //@begin executeBapi_TransactionCommit()     IWDMessageManager manager = wdComponentAPI.getMessageManager ();     try     {         wdContext.currentBapi_Transaction_Commit_InputElement().modelObject().execute();         wdContext.nodeBapi_Transaction_Commit_Input().invalidate();         wdContext.nodeOutput_1().invalidate();     } catch (ARFC2ModelExecuteException e){         wdComponentAPI.getMessageManager().reportException( e.getMessage());     }     //@end } </pre> <pre> public void executeBapi_TransactionCommit( ) {     //@begin executeBapi_TransactionCommit()     IWDMessageManager manager = wdComponentAPI.getMessageManager ();      try     {          wdContext.currentBapi_Transaction_Commit_InputElement(). modelObject().execute();          wdContext.nodeBapi_Transaction_Commit_Input().invalidate ();         wdContext.nodeOutput_1().invalidate();      } catch (ARFC2ModelExecuteException e){          wdComponentAPI.getMessageManager().reportException ( e.getMessage());     }     //@end } </pre>
91.	<p>Now, create the implementation code for method <b>addSalesOrderItem()</b>.</p> <p>SAVE.</p>	

```

public void addSalesOrderItem() {
    //@begin addSalesOrderItem()
    SOModel soModel = new SOModel();

    Bapisdhd1 header = new Bapisdhd1(soModel);
    Bapisdhd1X headerx = new Bapisdhd1X(soModel);
    Bapisditm orderItem = new Bapisditm(soModel);
    Bapisditmx orderItemx = new Bapisditmx(soModel);
    Bapisparnr partner = new Bapisparnr(soModel);
    Bapischdl schedule = new Bapischdl(soModel);
    Bapischdlx schedulx = new Bapischdlx(soModel);

    IPublicSOModelController.IOrder_Header_InElement eleh = wdContext.nodeOrder_Header_In().createOrder_Header_InElement(header);
    IPublicSOModelController.IOrder_Header_InxElement elehx = wdContext.nodeOrder_Header_Inx().createOrder_Header_InxElement(headerx);
    IPublicSOModelController.IOrder_Items_InElement elei = wdContext.nodeOrder_Items_In().createOrder_Items_InElement(orderItem);
    IPublicSOModelController.IOrder_Items_InxElement eleix = wdContext.nodeOrder_Items_Inx().createOrder_Items_InxElement(orderItemx);
    IPublicSOModelController.IOrder_PartnersElementelep = wdContext.nodeOrder_Partners().createOrder_PartnersElement(partner);
    IPublicSOModelController.IOrder_Schedules_InElement eles = wdContext.nodeOrder_Schedules_In().createOrder_Schedules_InElement(schedule);
    IPublicSOModelController.IOrder_Schedules_InxElement elesx = wdContext.nodeOrder_Schedules_Inx().createOrder_Schedules_InxElement(schedulx);

    String material = wdContext.currentContextElement().getMaterial();
    int qty = wdContext.currentContextElement().getQty();
    BigDecimal targetQty = new BigDecimal(qty);
    String uom = wdContext.currentContextElement().getUoM();
    int counterItem = wdContext.nodeOrder_Items_In().size();

    int item = 0;
    if (counterItem == 0) {
        item = 10;
    } else {
        item = (counterItem * 10) + 10;
    }

    String itemNumber2 = "0000" + item;

    elei.setItm_Number(itemNumber2);
    elei.setMaterial(material);
    elei.setTarget_Qty(targetQty);
    elei.setTarget_Qu(uom);

    eleix.setUpdateflag("X");
    eleix.setItm_Number(itemNumber2);
    eleix.setMaterial(true);
    eleix.setTarget_Qty(true);
    eleix.setTarget_Qu(true);

    eles.setItm_Number(itemNumber2);
    eles.setReq_Qty(targetQty);

    elesx.setUpdateflag("X");
    elesx.setItm_Number("000010");
    elesx.setReq_Qty(true);

    wdContext.nodeOrder_Items_In().addElement(elei);
    wdContext.nodeOrder_Items_Inx().addElement(eleix);
    wdContext.nodeOrder_Schedules_In().addElement(eles);
    wdContext.nodeOrder_Schedules_Inx().addElement(elesx);

    //@end
}

```

```

public void addSalesOrderItem( ) {
    //@@begin addSalesOrderItem()

    SOModel soModel = new SOModel();

    Bapisdhd1 header = new Bapisdhd1(soModel);
    Bapisdhd1X headerx = new Bapisdhd1X(soModel);
    Bapisditm orderItem = new Bapisditm(soModel);
    Bapisditmx orderItemx = new Bapisditmx(soModel);
    Bapiparnr partner = new Bapiparnr(soModel);
    Bapischdl schedule = new Bapischdl(soModel);
    Bapischdlx schedulex = new Bapischdlx(soModel);

    IPublicSOModelController.IOrder_Header_InElement eleh =
    wdContext.nodeOrder_Header_In().createOrder_Header_InEle
    ment(header);

    IPublicSOModelController.IOrder_Header_InxElement elehx =
    wdContext.nodeOrder_Header_Inx().createOrder_Header_InxE
    lement(headerx);
    IPublicSOModelController.IOrder_Items_InElement elei =
    wdContext.nodeOrder_Items_In().createOrder_Items_InEleme
    nt(orderItem);
    IPublicSOModelController.IOrder_Items_InxElement eleix =
    wdContext.nodeOrder_Items_Inx().createOrder_Items_InxEle
    ment(orderItemx);
    IPublicSOModelController.IOrder_PartnersElement elep =
    wdContext.nodeOrder_Partners().createOrder_PartnersEleme
    nt(partner);

    IPublicSOModelController.IOrder_Schedules_InElement eles
    =
    wdContext.nodeOrder_Schedules_In().createOrder_Schedules
    _InElement(schedule);

    IPublicSOModelController.IOrder_Schedules_InxElement
    elesx
    =
    wdContext.nodeOrder_Schedules_Inx().createOrder_Schedule
    s_InxElement(schedulex);

```

		<pre> //Set item number automatically based on data that user input  String material = wdContext.currentContextElement().getMaterial(); int qty = wdContext.currentContextElement().getQty();     BigDecimal targetQty = new BigDecimal(qty); String uom = wdContext.currentContextElement().getUoM();  int counterItem = wdContext.nodeOrder_Items_In().size();      int item = 0;     if (counterItem == 0){         item = 10;     }else{         item = (counterItem * 10) + 10;     }  String itemNumber2 = "0000" + item;  elei.setItm_Number(itemNumber2); elei.setMaterial(material); elei.setTarget_Qty(targetQty); elei.setTarget_Qu(uom);  eleix.setUpdateflag("X"); eleix.setItm_Number(itemNumber2); eleix.setMaterial(true); eleix.setTarget_Qty(true); eleix.setTarget_Qu(true);  eles.setItm_Number(itemNumber2); eles.setReq_Qty(targetQty);  elesx.setUpdateflag("X"); elesx.setItm_Number("000010"); elesx.setReq_Qty(true);  wdContext.nodeOrder_Items_In().addElement(elei); wdContext.nodeOrder_Items_Inx().addElement(eleix); wdContext.nodeOrder_Schedules_In().addElement(eles); wdContext.nodeOrder_Schedules_Inx().addElement(elesx);  //@@end } </pre>
92.	Implement method <b>deleteSalesOrderItem()</b> . SAVE.	<pre> public void deleteSalesOrderItem() {     //begin deleteSalesOrderItem()     int n = wdContext.nodeOrder_Items_In().size();     int leadSelected = wdContext.nodeOrder_Items_In().getLeadSelection();      for(int i = n - 1; i &gt;= 0; --i){         if (wdContext.nodeOrder_Items_In().isMultiSelected(i)    leadSelected == i){             wdContext.nodeOrder_Items_In().removeElement(wdContext.nodeOrder_Items_In().getElementAt(i));         }     }     //@@end } </pre>

```

public void deleteSalesOrderItem( ) {
    //@@begin deleteSalesOrderItem()
    int n = wdContext.nodeOrder_Items_In().size();
    int leadSelected =
wdContext.nodeOrder_Items_In().getLeadSelection();

    for(int i = n - 1; i >= 0; --i){
        if
(wdContext.nodeOrder_Items_In().isMultiSelected(i) ||
leadSelected == i){

wdContext.nodeOrder_Items_In().removeElement(wdContext.n
odeOrder_Items_In().getElementAt(i));

        }

    }
    //@@end
}

```

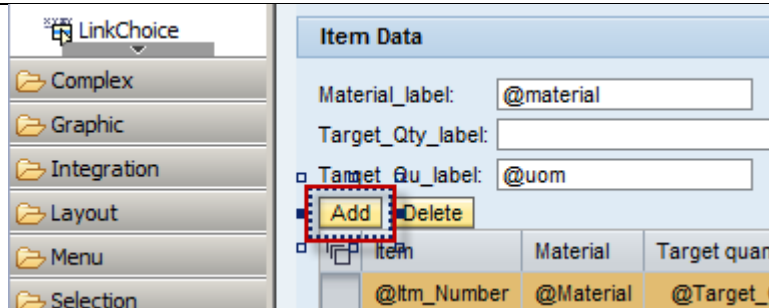
### STEP 13. CALL CUSTOM CONTROLLER METHOD ON EVENT BUTTON IN VIEW

93. Open the layout of SalesOrderView.

Double click on “Add” button to go in implementation menthod.

SAVE.

This code means that when “Add” button is clicked then the event will call method **onActionAddSalesOrderItem()**. Furthermore, it will call method **addSalesOrderItem** that already implemented in custom controller.



```

public void onActionAddSalesOrderItem(com.sap.tc.webdynpro.progmodel.api.IWDCustomEvent wdEvent )
{
    //@@begin onActionAddSalesOrderItem(ServerEvent)
    wdThis.wdGetSOModelControllerController().addSalesOrderItem();
    //@@end
}

```

```

public void
onActionAddSalesOrderItem(com.sap.tc.webdynpro.progmodel
.api.IWDCustomEvent wdEvent )
{
    //@@begin onActionAddSalesOrderItem(ServerEvent)

wdThis.wdGetSOModelControllerController().addSalesOrderI
tem();
    //@@end
}

```

94. Continue with “Delete” button.

SAVE.

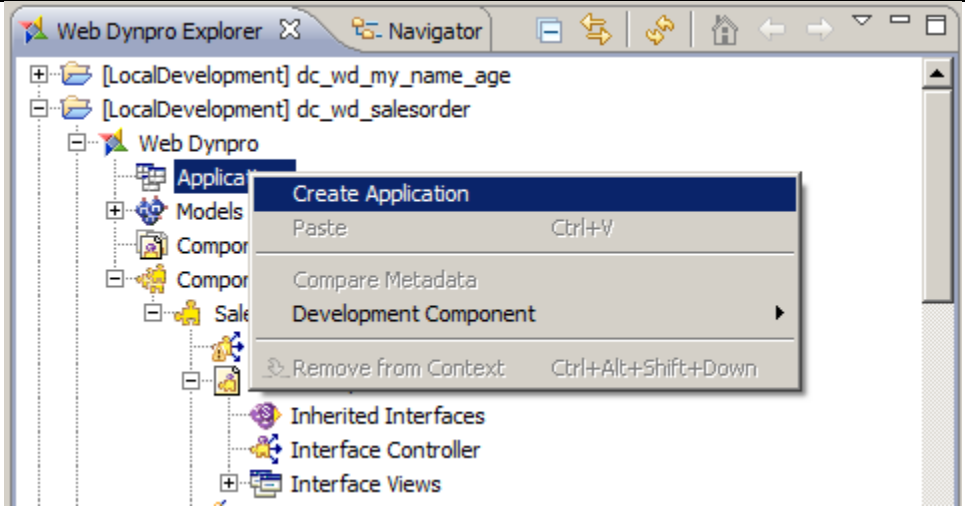
```

public void onActionDeleteItemSo(com.sap.tc.webdynpro.progmodel.api.IWDCustomEvent wdEvent )
{
    //@@begin onActionDeleteItemSo(ServerEvent)
    wdThis.wdGetSOModelControllerController().deleteSalesOrderItem();
    //@@end
}

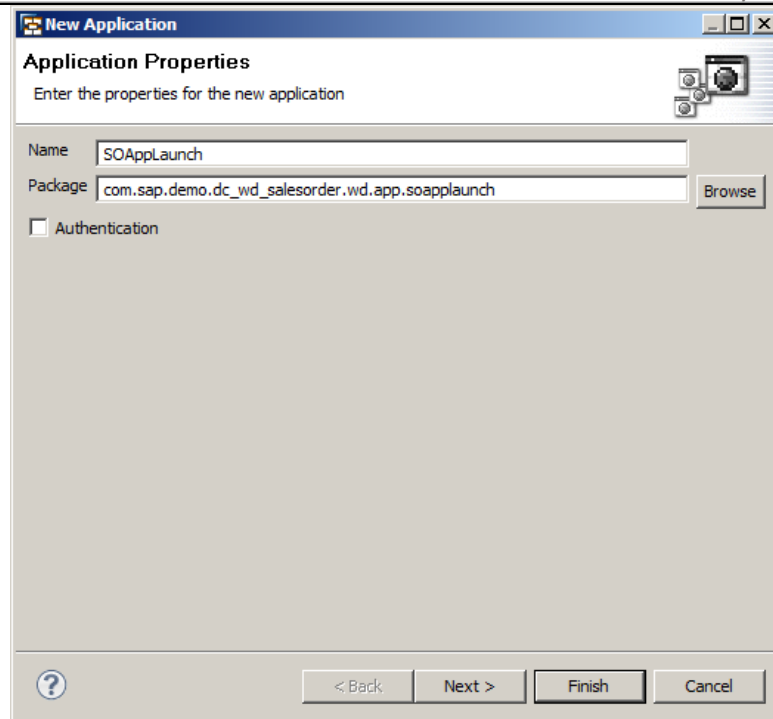
```

		<pre> public void onActionDeleteItemSo (com.sap.tc.webdynpro.progmodel.api. IWDCustomEvent wdEvent ) {     //@@begin onActionDeleteItemSo (ServerEvent)      wdThis.wdGetSOModelControllerController().deleteSalesOrd erItem();     //@@end } </pre>
95.	Finish with button "Create Sales Order".  SAVE.	<pre> public void onActionCreateSalesOrder (com.sap.tc.webdynpro.progmodel.api.IWDCustomEvent wdEvent ) {     //@@begin onActionCreateSalesOrder (ServerEvent)     wdThis.wdGetSOModelControllerController().executeBapi_Salesorder_Createfromdat2();     wdThis.wdGetSOModelControllerController().executeBapi_TransactionCommit();     //@@end } </pre> <pre> public void onActionCreateSalesOrder (com.sap.tc.webdynpro.progmodel. api.IWDCustomEvent wdEvent ) {     //@@begin onActionCreateSalesOrder (ServerEvent)      wdThis.wdGetSOModelControllerController().executeBapi_Sa lesorder_Createfromdat2();      wdThis.wdGetSOModelControllerController().executeBapi_Tr ansactionCommit();     //@@end } </pre>

## STEP 14. BUILDING, DEPLOYING, CONFIGURING, AND RUNNING YOUR APPLICATION

96.	Create application.  Right click on <i>Application &gt; Create          Application.</i>	 <p>The screenshot shows the Web Dynpro Explorer interface. The 'Web Dynpro' tree is expanded, and the 'Application' node is selected. A right-click context menu is displayed over the 'Application' node, showing the following options: 'Create Application' (highlighted), 'Paste' (with Ctrl+V shortcut), 'Compare Metadata', 'Development Component' (with a right-pointing arrow), and 'Remove from Context' (with Ctrl+Alt+Shift+Down shortcut). The background shows the project structure with nodes like 'Models', 'Components', 'Sales', 'Inherited Interfaces', 'Interface Controller', and 'Interface Views'.</p>
-----	-------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

A new window appear,  
type "SOAppLaunch".  
Click Next.



**New Application**

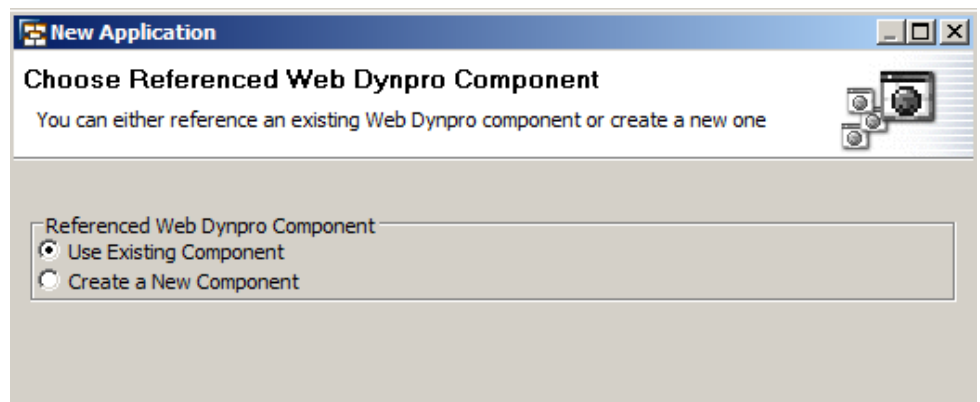
**Application Properties**  
Enter the properties for the new application

Name:

Package:

☐ Authentication

Choose *Use Existing Component*, click  
Next.



**New Application**

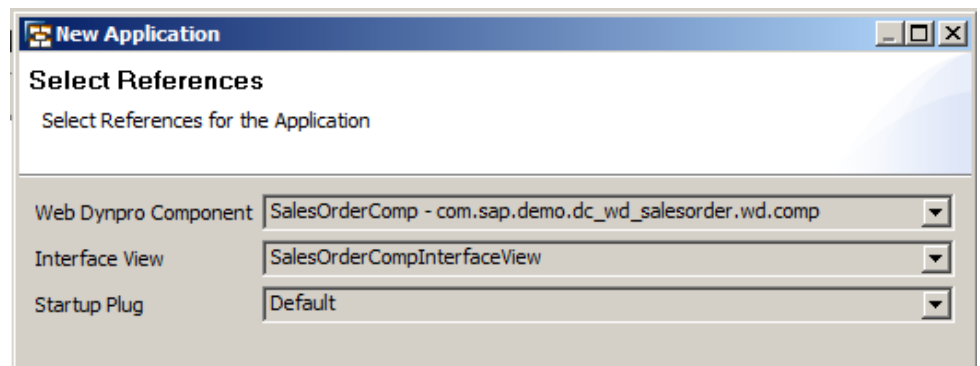
**Choose Referenced Web Dynpro Component**  
You can either reference an existing Web Dynpro component or create a new one

☐ Referenced Web Dynpro Component

☒ Use Existing Component

☐ Create a New Component

Click Finish.



**New Application**

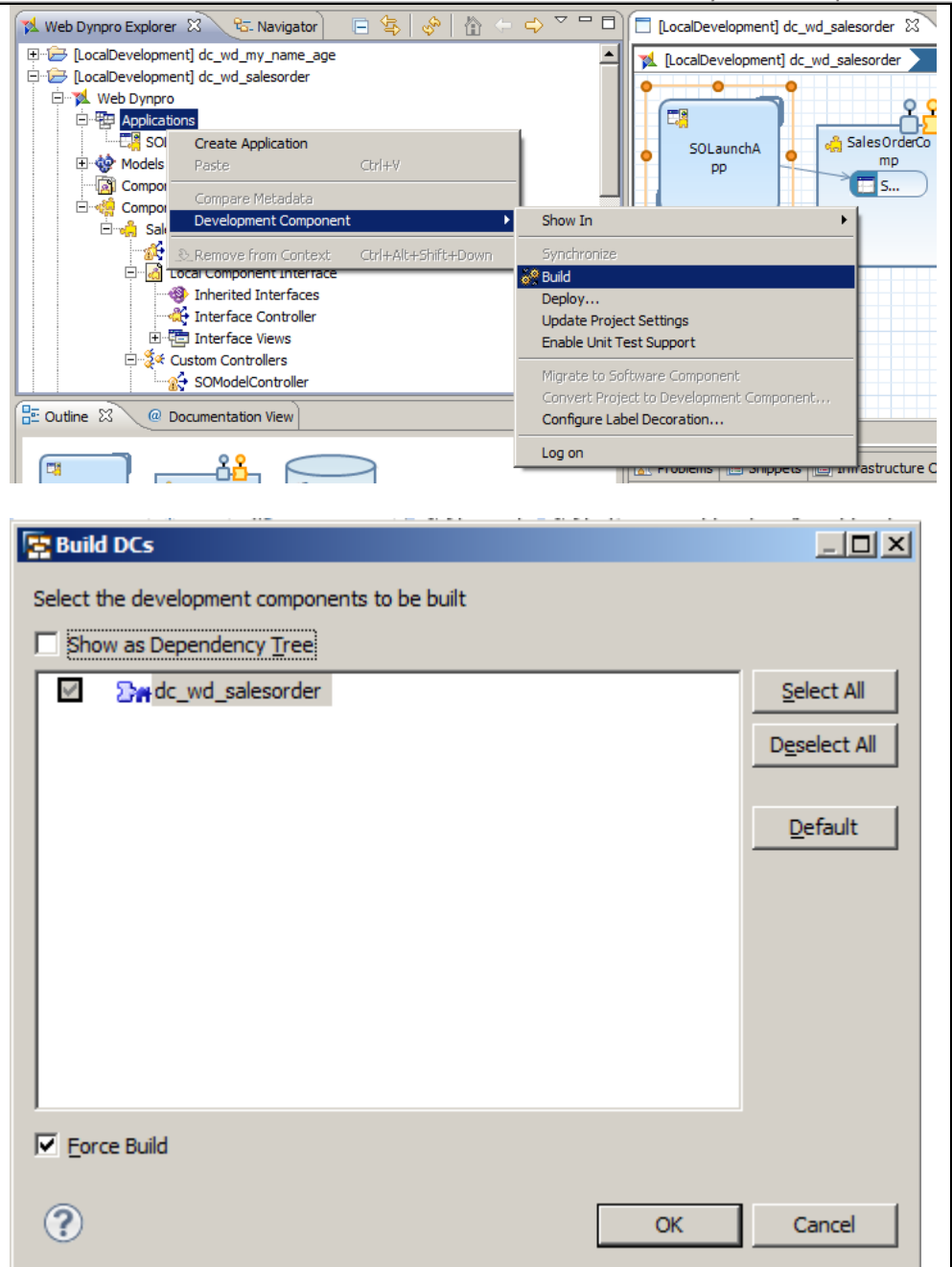
**Select References**  
Select References for the Application

Web Dynpro Component:

Interface View:

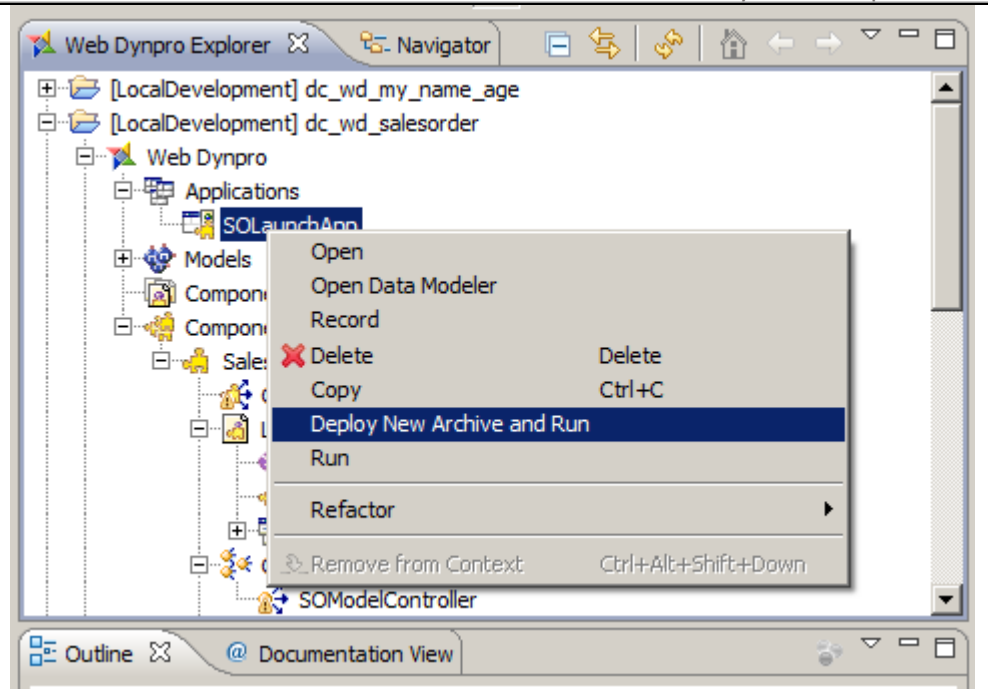
Startup Plug:

97. Building the project.  
Click Finish.





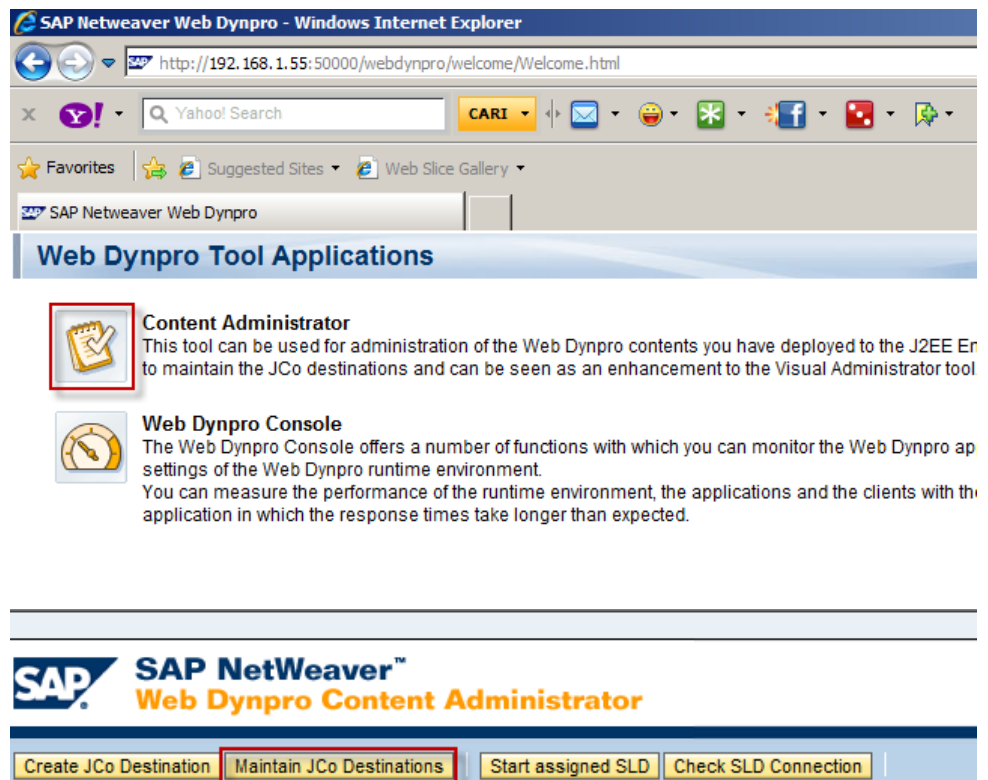
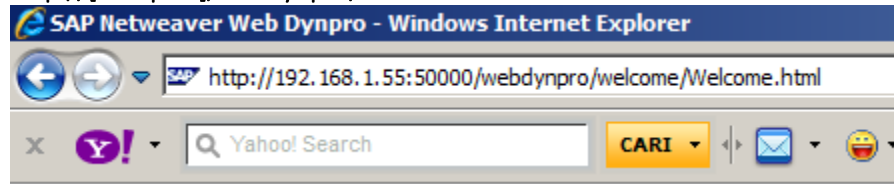
98. Deploying the project. Right click on SOLaunchApplication > *Deploy New Archive and Run.*



99. Creating JCO Connection in Content Administrator.

Open your NWA Content Administrator:

[http://\[URL:port\]/webdynpro/welcome](http://[URL:port]/webdynpro/welcome)



**JCo Destination Details**

Define Source:

Name
demo.sap.com/createsalesorder
demo.sap.com/createso
demo.sap.com/createso_app
demo.sap.com/dc_wd_jeff_name_age
demo.sap.com/dc_wd_my_name_age
demo.sap.com/dc_wd_my_name_age_2
demo.sap.com/dc_wd_my_name_age3
demo.sap.com/dc_wd_salesorder
demo.sap.com/dc_wd_so
demo.sap.com/dc_wd_so2

Create new JCo Destination

1 General Data    2 J2EE Cluster    2 Connection Type    3.1 Appl. Server Connection

---

**JCo Destination Details**

Define Source:

Name	Start
Create a new JCo destination using the creation w	

### Create RFC Application Data:

**Create new JCo Destination**

1 General Data    2 J2EE Cluster    2 Connection Type    3.1 Appl. Server Connection    3.2 Msg. Server Connection    4 Security    5 Summary

Define the name and the maximal pool size of the JCo connection.  
Optionally you can create the new JCo destination as a copy of an existing one.

Destination Name	JCo Pool Configuration
Name: <input type="text" value="WD_SO_MODELDATA_DEST"/>	Maximal Pool Size: <input type="text" value="5"/>
Client: <input type="text" value="800"/>	Maximum Connections: <input type="text" value="10"/>
<input type="checkbox"/> Copy it from an existing JCo destination	Connection Timeout (msec.): <input type="text" value="10,000"/>
<input type="text" value="WD_FLIGHTBOOKING_MODELDATA"/>	Maximum Waiting Time (msec.): <input type="text" value="30,000"/>

---

**Create new JCo Destination**

1 General Data    2 J2EE Cluster    2 Connection Type    3.1 Appl. Server Connection    3.2 Msg. Server Connection    4 Security    5 Summary

Define the J2EE cluster, to which the JCo destination should be assigned.  
You should use your local J2EE cluster as default. In some cases it might be useful to define the JCo destination for another J2EE cluster.

☒ Use local J2EE engine "NTW on netweaver"

---

**Create new JCo Destination**

1 General Data    2 J2EE Cluster    2 Connection Type    3.1 Appl. Server Connection    3.2 Msg. Server Connection    4 Security    5 Summary

Data Type	Destination Type
Define the type of data you want to read using the JCo destination.	Define whether or not you want a load-balanced access.
<input type="radio"/> Dictionary Meta Data	<b>Note: This is only possible for destinations used to read application data.</b>
<input checked="" type="radio"/> Application Data	<input checked="" type="radio"/> Load-balanced Connection (recommended)
	<input type="radio"/> Single Server Connection (should be used only for debugging)

**Create new JCo Destination**

1 General Data 2 J2EE Cluster 2 Connection Type 3.1 Appl. Server Connection 3.2 Msg. Server Connection 4 Security 5 Summary

Define the message server, system name and the logon group used by the JCO connection.

Message Server:

☐ Enter Host name manually

Message Server Host Name:

System Name:

Logon Group:

☐ Use SAP Router

SAP Router String:

[Previous](#) [Next](#) [Finish](#) [Cancel](#)

**Create new JCo Destination**

1 General Data 2 J2EE Cluster 2 Connection Type 3.1 Appl. Server Connection 3.2 Msg. Server Connection 4 Security 5 Summary

Define the required authentication method and (optionally) the parameters needed for a secure network communication (SNC).

Define the used authentication method and (optionally) the parameters needed for a secure network connection (SNC). We recommend in general to use assertion ticket as preferred to SSO ticket.

**User Authentication**

Used Method:

Name:

Password:

Confirm Password:

Language:

**Secure Network Connection (SNC)**

SNC Mode:

SNC Partner Name:

SNC Security Level:

SNC Name:

SNC Library Path:

[Previous](#) [Next](#) [Finish](#) [Cancel](#)

**Create new JCo Destination**

1 General Data 2 J2EE Cluster 2 Connection Type 3.1 Appl. Server Connection 3.2 Msg. Server Connection 4 Security 5 Summary

You defined the following JCO connection:

**General** **Security** **Connection**

**General Data**

Name:

Client:

J2EE Cluster Name:

**JCo Pool Configuration**

Maximum Pool Size:

Maximum Connections:

Connection Timeout (sec.):

Maximum Waiting Time (sec.):

[Previous](#) [Next](#) [Finish](#) [Cancel](#)

### Test connection:

Name	Status	Create	Preview	Edit	Test	Ping	Remove
WD_FLIGHTLIST_RFC_METADATA_DEST	✓	Create	Preview	Edit	Test	Ping	Remove
WD_MODELDATA_DEST	✓	Create	Preview	Edit	Test	Ping	Remove
WD_NEWMODELDATA_DEST	✓	Create	Preview	Edit	Test	Ping	Remove
WD_RFC_METADATA_DEST	✓	Create	Preview	Edit	Test	Ping	Remove
WD_SO_MODELDATA_DEST	✓	Create	Preview	Edit	Test	Ping	Remove

☒ JCo destination 'WD\_SO\_MODELDATA\_DEST' was successfully tested with user 'PST\_MONICA'

### Create RFC Data Dictionary:

**Create new JCo Destination**

1 2 2 3.1 3.2 4 5  
General Data J2EE Cluster Connection Type Appl. Server Connection Msg. Server Connection Security Summary

Define the name and the maximal pool size of the JCo connection.  
Optionally you can create the new JCo destination as a copy of an existing one.

Destination Name	JCo Pool Configuration
Name: <input type="text" value="WD_SO_RFC_METADATA_DEST"/>	Maximal Pool Size: <input type="text" value="5"/>
Client: <input type="text" value="800"/>	Maximum Connections: <input type="text" value="10"/>
<input type="checkbox"/> Copy it from an existing JCo destination	Connection Timeout (msec.): <input type="text" value="10,000"/>
<input type="text" value="WD_FLIGHTBOOKING_MODELDA1"/>	Maximum Waiting Time (msec.): <input type="text" value="30,000"/>

Previous Next Finish Cancel

**Create new JCo Destination**

1 2 2 3.1 3.2 4 5  
General Data J2EE Cluster Connection Type Appl. Server Connection Msg. Server Connection Security Summary

Define the J2EE cluster, to which the JCo destination should be assigned.  
You should use your local J2EE cluster as default. In some cases it might be useful to define the JCo destination for another J2EE cluster.

☒ Use local J2EE engine "NTW on netweaver"

Previous Next Finish Cancel

**Create new JCo Destination**

1 2 2 3.1 3.2 4 5  
General Data J2EE Cluster Connection Type Appl. Server Connection Msg. Server Connection Security Summary

**Data Type**

Define the type of data you want to read using the JCo destination.

☒ Dictionary Meta Data  
☐ Application Data

**Destination Type**

Define whether or not you want a load-balanced access.  
**Note: This is only possible for destinations used to read application data.**

☒ Load-balanced Connection (recommended)  
☐ Single Server Connection (should be used only for debugging)

Previous Next Finish Cancel

**Create new JCo Destination**

1 2 2 3.1 3.2 4 5  
General Data J2EE Cluster Connection Type Appl. Server Connection Msg. Server Connection Security Summary

Define the message server, system name and the logon group used by the JCo connection.

Message Server:   
☐ Enter Host name manually

Message Server Host Name:

System Name:

Logon Group:   
☐ Use SAP Router

SAP Router String:

Previous Next Finish Cancel

**Create new JCo Destination**

1 2 2 3.1 3.2 4 5  
General Data J2EE Cluster Connection Type Appl. Server Connection Msg. Server Connection Security Summary

Define the required authentication method and (optionally) the parameters needed for a secure network communication (SNC).  
Define the used authentication method and (optionally) the parameters needed for a secure network connection (SNC).  
We recommend in general to use assertion ticket as preferred to SSO ticket.

User Authentication	Secure Network Connection (SNC)
Used Method: <input type="text" value="User / Password"/>	SNC Mode: <input type="text" value="Off"/>
Name: <input type="text" value="PST_MONICA"/>	SNC Partner Name: <input type="text"/>
Password: <input type="password" value="*****"/>	SNC Security Level: <input type="text"/>
Confirm Password: <input type="password" value="*****"/>	SNC Name: <input type="text"/>
Language: <input type="text" value="English"/>	SNC Library Path: <input type="text"/>

Previous Next Finish Cancel

**Create new JCo Destination**

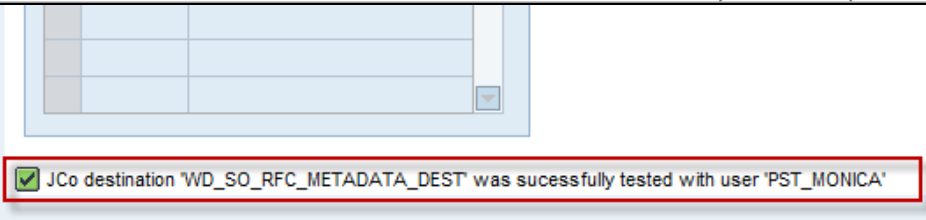
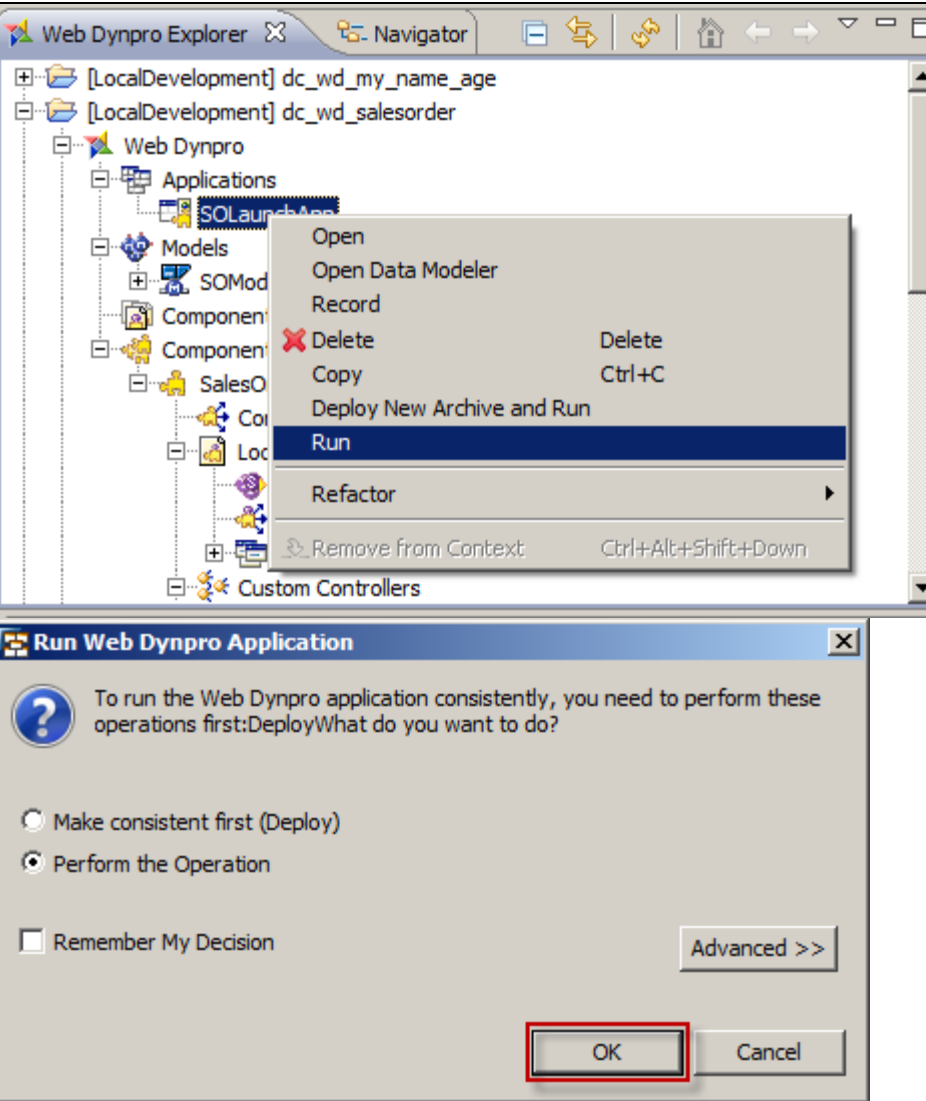
1 2 2 3.1 3.2 4 5  
General Data J2EE Cluster Connection Type Appl. Server Connection Msg. Server Connection Security Summary

You defined the following JCo connection:

General Security Connection

General Data	JCo Pool Configuration
Name: <input type="text" value="WD_SO_RFC_METADATA_DEST"/>	Maximum Pool Size: <input type="text" value="5"/>
Client: <input type="text" value="800"/>	Maximum Connections: <input type="text" value="10"/>
J2EE Cluster Name: <input type="text" value="NTW on netweaver"/>	Connection Timeout (sec.): <input type="text" value="10,000"/>
	Maximum Waiting Time (sec.): <input type="text" value="30,000"/>

Previous Next Finish Cancel

		 <p>✓ JCo destination 'WD_SO_RFC_METADATA_DEST' was successfully tested with user 'PST_MONICA'</p>
100.	Launching the application.	 <p>Web Dynpro Explorer</p> <p>Navigator</p> <p>[LocalDevelopment] dc_wd_my_name_age</p> <p>[LocalDevelopment] dc_wd_salesorder</p> <p>Web Dynpro</p> <p>Applications</p> <p>SOLaunchApp</p> <p>Models</p> <p>SOMod</p> <p>Component</p> <p>Component</p> <p>SalesO</p> <p>Col</p> <p>Loc</p> <p>Custom Controllers</p> <p>Open</p> <p>Open Data Modeler</p> <p>Record</p> <p>Delete</p> <p>Delete</p> <p>Copy</p> <p>Ctrl+C</p> <p>Deploy New Archive and Run</p> <p>Run</p> <p>Refactor</p> <p>Remove from Context</p> <p>Ctrl+Alt+Shift+Down</p> <p>Run Web Dynpro Application</p> <p>To run the Web Dynpro application consistently, you need to perform these operations first:DeployWhat do you want to do?</p> <p><input type="radio"/> Make consistent first (Deploy)</p> <p><input checked="" type="radio"/> Perform the Operation</p> <p><input type="checkbox"/> Remember My Decision</p> <p>Advanced &gt;&gt;</p> <p>OK</p> <p>Cancel</p>

101. Result.

Since, we not implemented yet data conversion, data such as order type is still using default language that acknowledged by BAPI.

SOLaunchApp

### CREATE SALES ORDER

**Header Data**

Doc\_Type\_label:

Sales\_Org\_label:

Distr\_Chanel\_label:

Division\_label:

Purch\_No\_C\_label:

Partn\_Numb\_label:

**Item Data**

Qty:  0

UoM:

Material:

**Add Delete**

Target quantity	Material	UoM	Item

**Create Sales Order**

Fid.	Message	MsgType	Msg.No.	Message Variable	Message Variable	Message Variable	Msg.no	Log number

Salesdocument\_label:

### CREATE SALES ORDER

**Header Data**

Doc\_Type\_label:

Sales\_Org\_label:

Distr\_Chanel\_label:

Division\_label:

Purch\_No\_C\_label:

Partn\_Numb\_label:

**Item Data**

Qty:  6

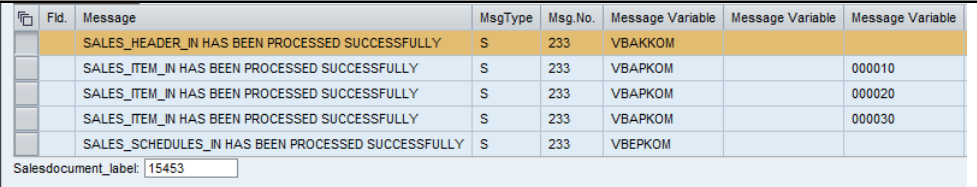
UoM:

Material:

**Add Delete**

Target quantity	Material	UoM	Item
4	T-ATA20	ST	000010
2	T-ATA20	ST	000020
6	T-ATA20	ST	000030

**Create Sales Order**

		 <p>Salesdocument_label: 15453</p>
102.	<p>Checking the result.</p> <p>Based on the Sales Order number that created in Java Web Dynpro , we then checked the data in the SAP backend system.</p> <p>Now we already succeed to create simple application Create Sales Order in Java Web Dynpro and recorded automatically in SAP backend system.</p>	<p>In SAP backend system, open t-code VA03 display Sales Order.</p> 