

## Mapping between ASBOs and GBOs

What this exercise is about .....	2
Lab requirements .....	2
What you should be able to do .....	2
Introduction .....	3
Description of the module.....	4
Exercise instructions .....	5
Part 1: Initialize the workspace for this lab exercise .....	7
Part 2: Define interface maps and data maps for PeopleSoft.....	9
Part 3: Define interface maps and data maps for Siebel .....	27
Part 4: Test the PeopleSoft maps.....	36
Part 5: Test the Siebel maps.....	43
What you did in this exercise .....	50
Solution instructions .....	51
Task: Adding remote server to WebSphere Integration Developer test environment .....	52

---

## What this exercise is about

This exercise shows you how to develop Interface maps (mediations) and Data maps. These are used to map between the PeopleSoft ASBOs and the GBOs and from the GBOs to the Siebel ASBOs.

## Lab requirements

List of system, software and other tasks required for the student to complete the lab.

- WebSphere Integration Developer V6.0.2 installed
- WebSphere Process Server V6.0 test environment installed
- Sample code in the directory C:\Labfiles602 (Windows®) or /tmp/LabFiles602 (Linux®)
- IBM WebSphere Business Integration Toolset installed
- Visual Test Connector

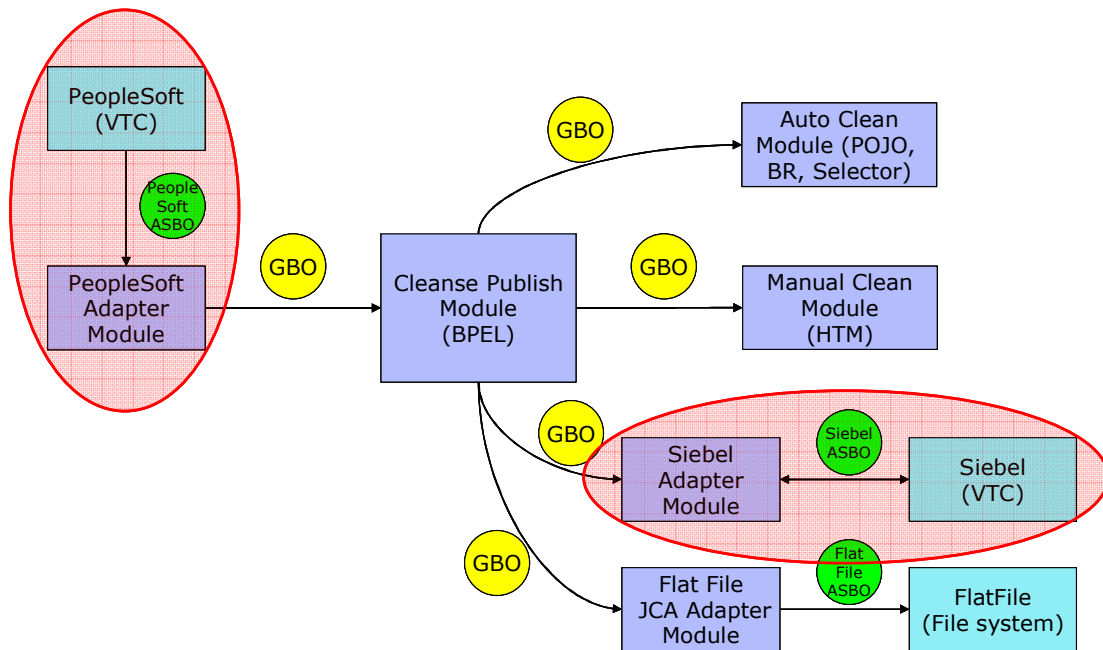
## What you should be able to do

At the end of this lab you should be able to

- Create and Interface map and define the handling of the parameters for the interfaces
- Create a data map with a variety of transforms
- Test the maps using Component Test

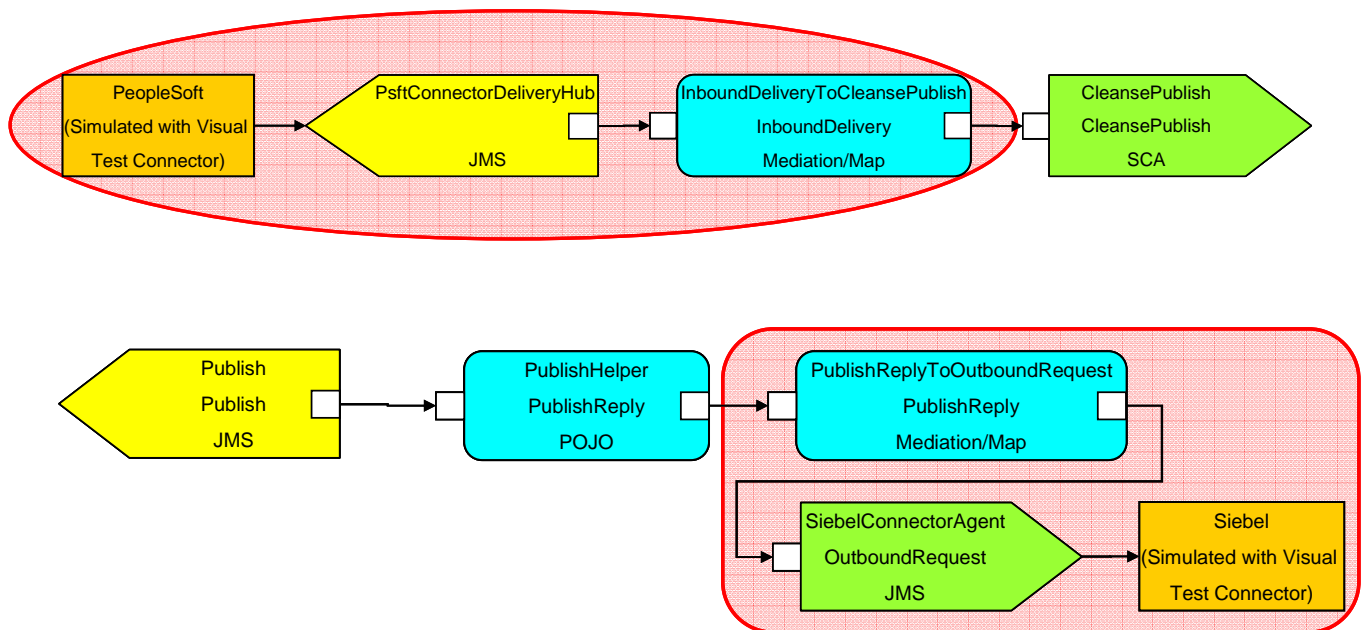
## Introduction

The following diagram highlights the parts of the overall scenario that will be addressed in this lab. You will add mapping of business objects to the PeopleSoft and Siebel modules. This will build upon what you have already developed for these modules in the **WebSphere Business Integration adapters – Import and export** lab.



## Description of the module

Data mapping operations are used to convert between ASBOs and GBOs. In WebSphere Integration Developer, Data Maps are always contained within Interface Maps. What you will do is add an Interface Map to each module, and then build the Data Maps that are required to do the PeopleSoft ASBO to GBO mappings and the GBO to Siebel ASBO mappings. Then the test component will be used to test the Interface Map and its contained Data Maps.



---

## Exercise instructions

Some instructions in this lab might be specific for Windows platforms. If you run the lab on a platform other than Windows, you will need to run the appropriate commands, and use appropriate files ( for example .sh in place of .bat) for your operating system. The directory locations are specified in the lab instructions using symbolic references as follows:

Reference variable	Windows location	Linux location
<WID_HOME>	C:\Program Files\IBM\WebSphere\ID\6.0	/opt/IBM/WebSphere/ID/6.0
<WPS_HOME>	<WID_HOME>\runtimes\bi_v6	<WID_HOME>/runtimes/bi_v6
<LAB_FILES>	C:\Labfiles602	/tmp/Labfiles602
<WORKSPACE>	C:\Labfiles602\eXchange\WBIAdapters\workspaceMaps	/tmp/Labfiles602/eXchange/WBIAdapters\workspace
<TEMP>	C:\temp	/tmp
<SOLUTION>	C:\Labfiles602\WBIAdapters\Solution	/tmp/Labfiles602\WBIAdapters\Solution

**Windows users' note:** When directory locations are passed as parameters to a Java™ program such as wsadmin, you must replace the backslashes with forward slashes to follow the Java convention. For example, C:\LabFiles602\ would be replaced by C:/LabFiles602/.

Note that the previous table is relative to where you are running WebSphere Integration Developer. The following table is related to where you are running remote test environment:

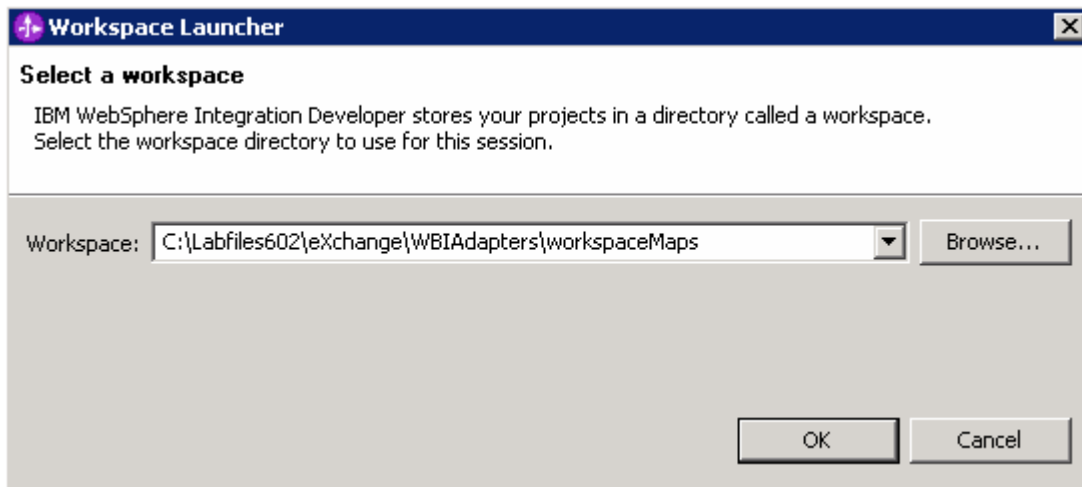
Reference Variable	Example: Remote Windows test server location	Example: Remote z/OS <sup>®</sup> test server location	Input your values for the remote location of the test server
<SERVER_NAME>	server1	cl1sr01	
<WAS_HOME>	C:\Program Files\IBM\WebSphere\AppServer	/etc/cl1cell/AppServerNode1	
<HOSTNAME>	localhost	mvsxxx.rtp.raleigh.ibm.com	
<BOOTSTRAP_PORT>	2809	2809	
<TELNET_PORT>	N/A	1023	
<PROFILE_NAME>	AppSrv01	default	
<USERID>	N/A	cl1admin	
<PASSWORD>	N/A	fr1day	

Instructions for using a remote testing environment, such as z/OS<sup>®</sup>, AIX<sup>®</sup> or Solaris, can be found at the end of this document, in the section "[Task: Adding remote server to WebSphere Integration Developer test environment](#)".

## Part 1: Initialize the workspace for this lab exercise

This lab exercise is dependent upon the completion of the **WBIAdapters – Import/Export** lab exercise. There are two approaches you can take to performing this lab.

- \_\_\_ 1. Start WebSphere Integration Developer V6.0.2 with a workspace location of **<WORKSPACE>**, that is **<LAB\_FILES>\eXchange\WBIAdapters\workspaceMaps**
  - \_\_\_ a. From Windows Explorer, navigate to the **<WID\_HOME>** directory and double click on wid.exe
  - \_\_\_ b. When prompted for workspace, enter the value provided by the **<WORKSPACE>** variable for this lab and click **OK**



- \_\_\_ c. When WebSphere Integration Developer V6.0.2 opens, click the curved arrow at top right to **go to Business Integration perspective**



- \_\_\_ d. Ensure you are in the **Business Integration** perspective
- \_\_\_ 2. Import Project Interchange file, **ImportExport\_P1.zip** located at **<LAB\_FILES>\eXchange\WBIAdapters\Solution\**

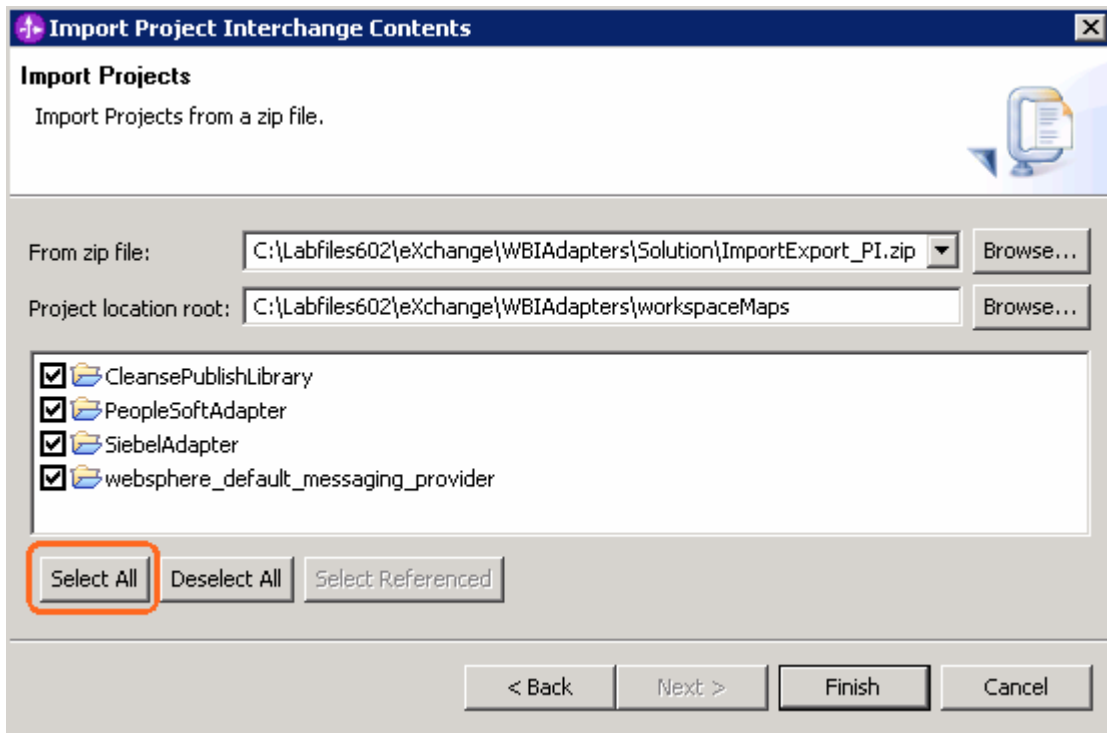
---

**Note:** As this lab is dependent upon **WBIAdapters – Import/Export** lab exercise, the resultant of its solution will be used as an import in the current lab.

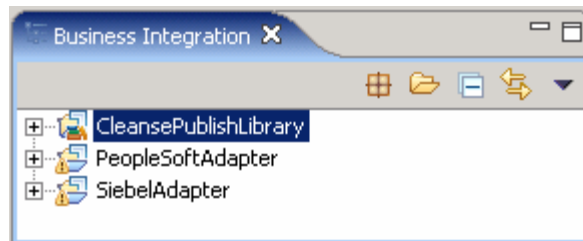
---

- \_\_\_ a. Right-click inside **Business Integration View** (top left view in the Business Integration Perspective) and select **Import** from the context menu
- \_\_\_ b. Select **Project Interchange** listed in the import dialog

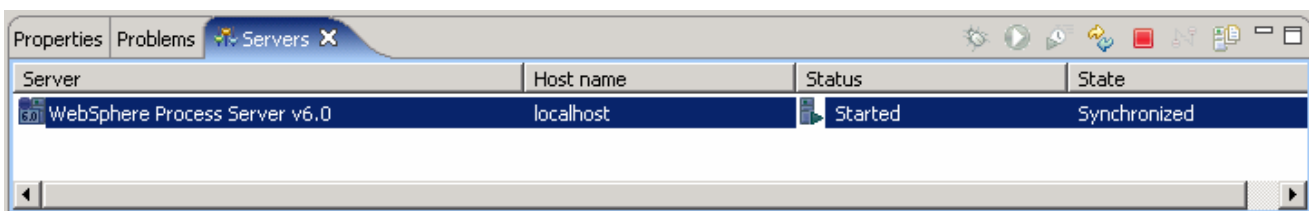
- \_\_\_ c. Click **Next**
- \_\_\_ d. Click the **Browse** button for “**From zip file**” and navigate to **<LAB\_FILES>\eXchange\WBIAdapters\Solution\ImportExport\_PI.zip** and hit **Open**



- \_\_\_ e. Click the **Select All** button and Click **Finish**
- \_\_\_ f. Verify that you have **CleansePublishLibrary**, **PeopleSoftAdapter** and **SiebelAdapter** modules are listed in the Business Integration view




- \_\_\_ g. Verify you have WebSphere Process Server V6.0 listed in your Servers view






## Part 2: Define interface maps and data maps for PeopleSoft

In this part you will create the Interface Map needed to map the InboundDelivery interface of the Export for the PeopleSoft adapter to the CleansePublish interface which will be called on the CleansePublish module. Within the Interface Map you will define the Data Maps required to map the PeopleSoft ASBOs to the GBOs.

\_\_\_ 1. Expand **PeopleSoftAdapter** in the Business Integration view and double click on **Assembly Diagram** (  **Assembly Diagram** ) to open it in an Assembly Diagram editor


\_\_\_ a. Add the Import that will be used to call the **CleansePublish** module

1) Drag an import onto the Assembly diagram

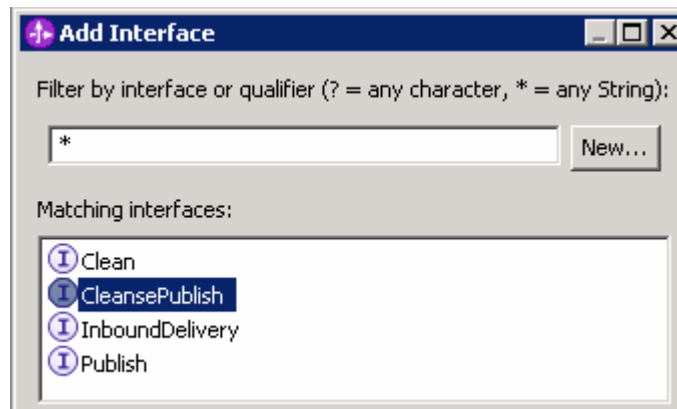
- a) Click on **Import** icon from Assembly Diagram tray (  )
- b) Click or drag import to the Assembly diagram



2) Hover over import or click on import to make the add interface icon appear.

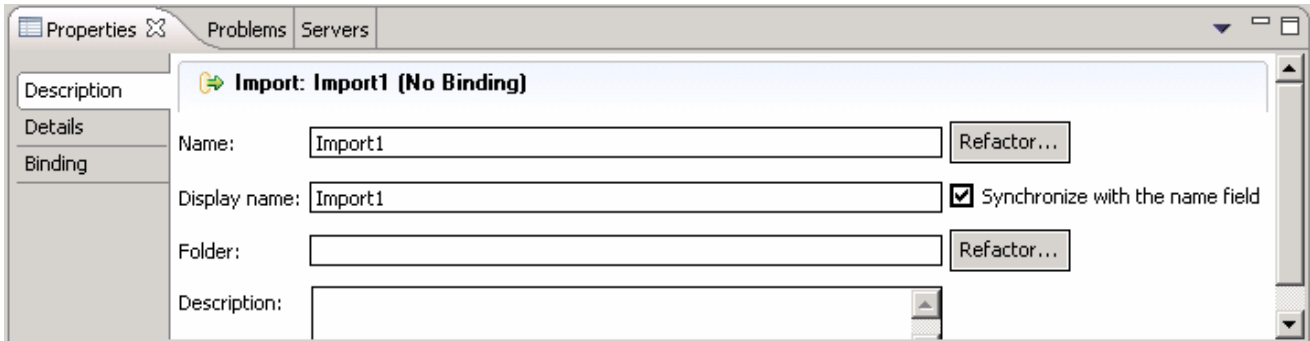
3) Click "I" icon (  ) to add an interface

4) Select **CleansePublish** from the Add Interface dialog and click **OK**



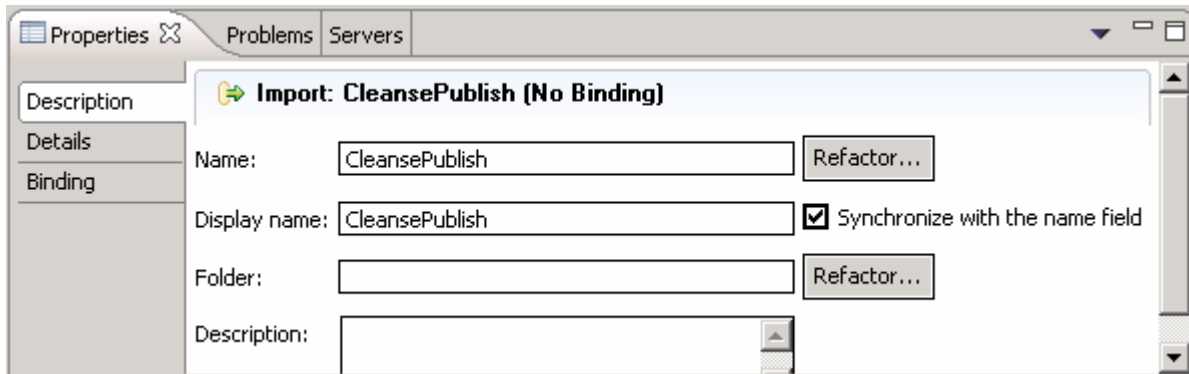
\_\_\_ b. Refactor the Description for the import, **Import1**

- 1) Select the Import, **Import1** on the Assembly Diagram
- 2) In the Properties view select the **Description** tab



3) Rename the **Display name** and **Name** to **CleansePublish**

**Hint:** Modifying Name field automatically synchronizes with the Display name with the name filed when the check box next to “Synchronize with the name filed” is selected.



\_\_ c. Generate SCA bindings for Import, **CleansePublish**

- 1) Right-click on CleansePublish and select **Generate Binding → SCA Binding**
- 2) This is how your Assembly Diagram should now look like:

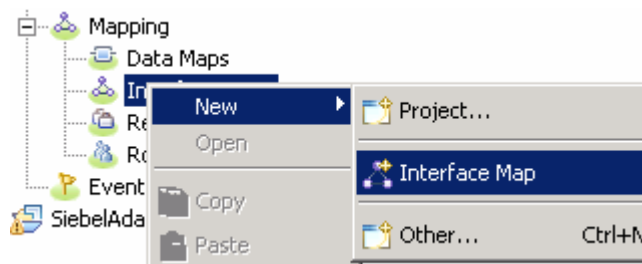


\_\_ d. Save all work by **File → Save All** or **Ctrl + Shift + S**

\_\_\_ 2. Add an Interface Map that will map the InboundDelivery interface to the CleansePublish interface

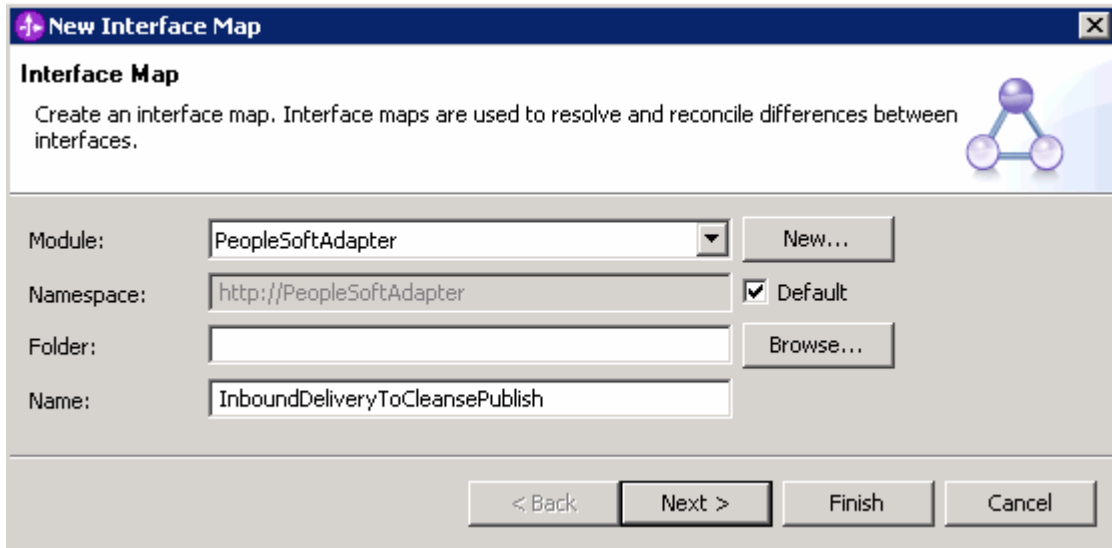
\_\_ a. In the Business Integration view, expand **PeopleSoftAdapter → Mapping**

\_\_ b. Right click on **Interface Maps** and select **New → Interface Map** from the context menu



\_\_ c. In the New Interface Map dialog, enter the following values:

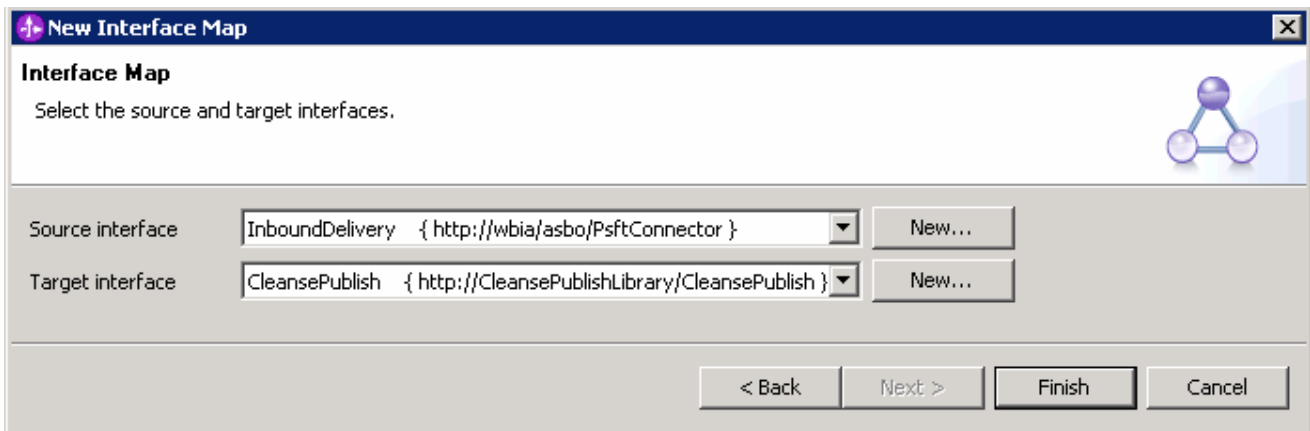
- 1) Module: **PeopleSoftAdapter**
- 2) Namespace: (accept the default)
- 3) Folder: (accept the default)
- 4) Name: **InboundDeliveryToCleansePublish**



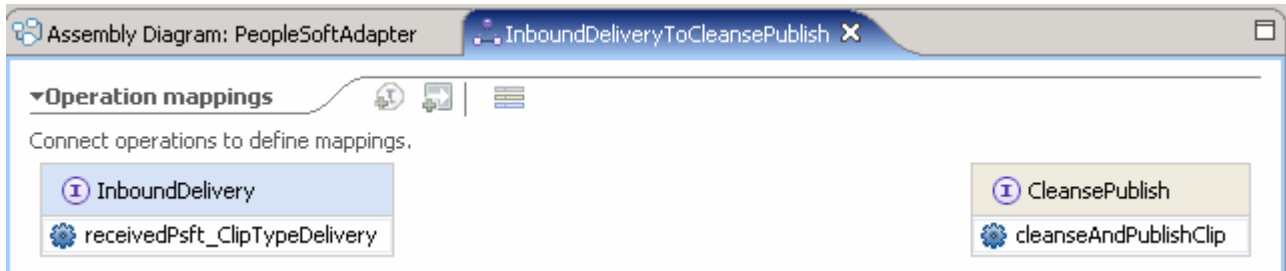
\_\_ d. Click **Next**

\_\_ e. In the following panel, select the following values:

- 1) Source Interface : **Inbound Delivery**
- 2) Target Interface : **CleansePublish**



\_\_ f. Click **Finish**. The Interface Mapping editor opens with the Operation Mappings pane visible as shown below:



3. Define the InboundDeliveryToCleansePublish Interface Map

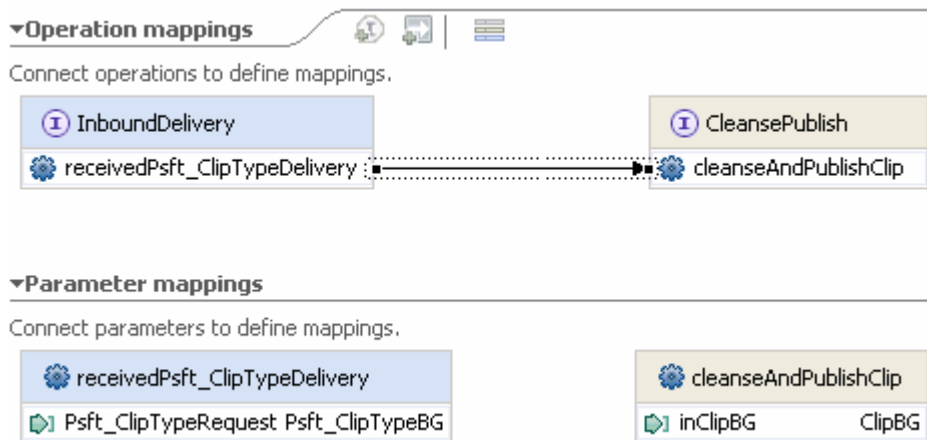
- a. Click on **receivedPsft\_ClipTypeDelivery** operation of the **InboundDelivery** Interface on the left-hand side of the Operation Connections view



- b. Drag to the **cleanseAndPublishClip** operation of the **CleansePublish** Interface on the right-hand side of the Operation Connections view and unclick. There should be a **black line** (wire) connecting **InboundDelivery** to **CleansePublish** Interfaces



- c. Click on the line (wire) between the two operations to view the Parameter mappings pane as shown below:



- d. Under the Parameter mappings section, draw a line from the **Psft\_ClipTypeRequest** parameter to the **inClipBG** parameter



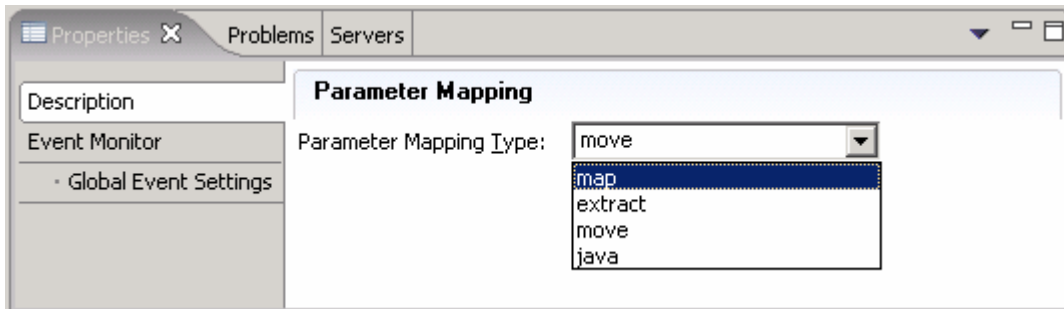
**Note:** Notice that the default parameter mapping operation is a **move**. Since these parameters are of different types, you will need to change this to a map in the next step.

\_\_\_ 4. Create the Data Map to **map** the Psft\_ClipTypeRequest parameter (which is a Psft\_ClipTypeBG) to the inClipBG parameter (which is a ClipBG)

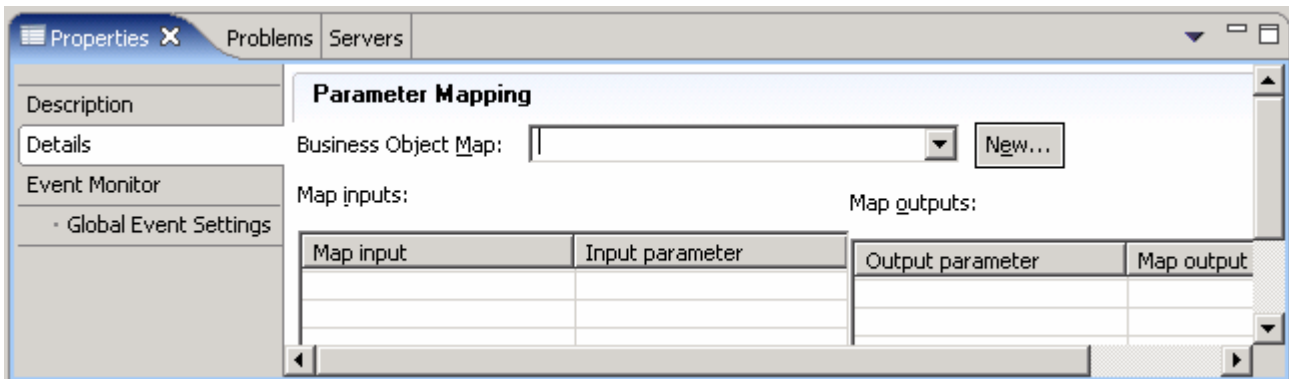
\_\_\_ a. Click on the wire (  ) to select it

\_\_\_ b. While the wire is selected, select the Description tab of the Properties view

\_\_\_ c. Use the drop down box to change the Parameter Mapping Type from move to **map**



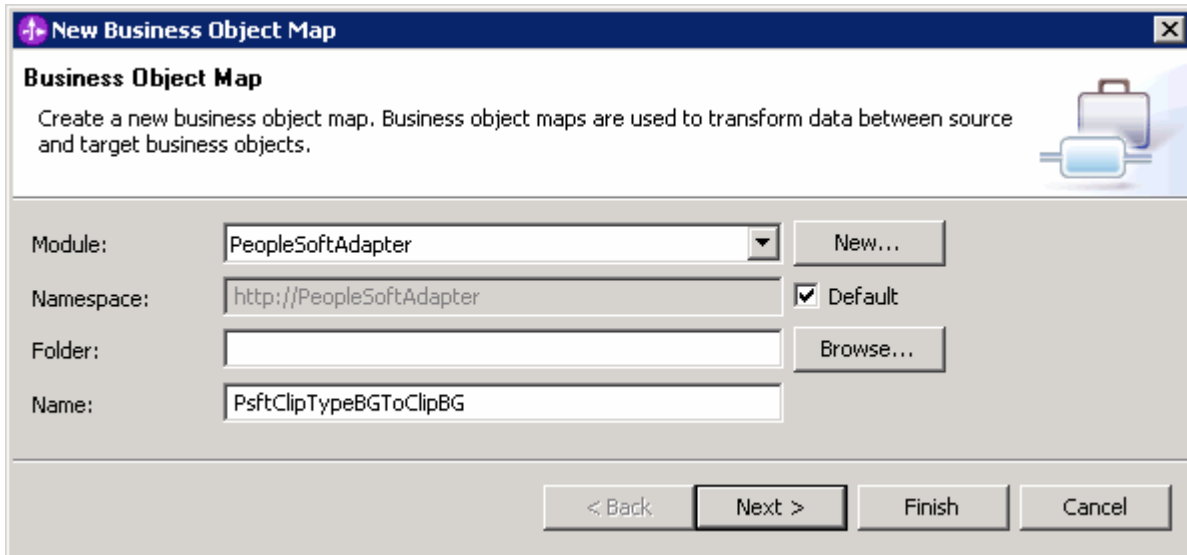
\_\_\_ d. Now click on the **Details** tab in the Properties view



\_\_\_ e. Click the **New...** button to open the New Business Object Mapping dialog

\_\_\_ f. In the New Business Object Mapping dialog, enter following information:

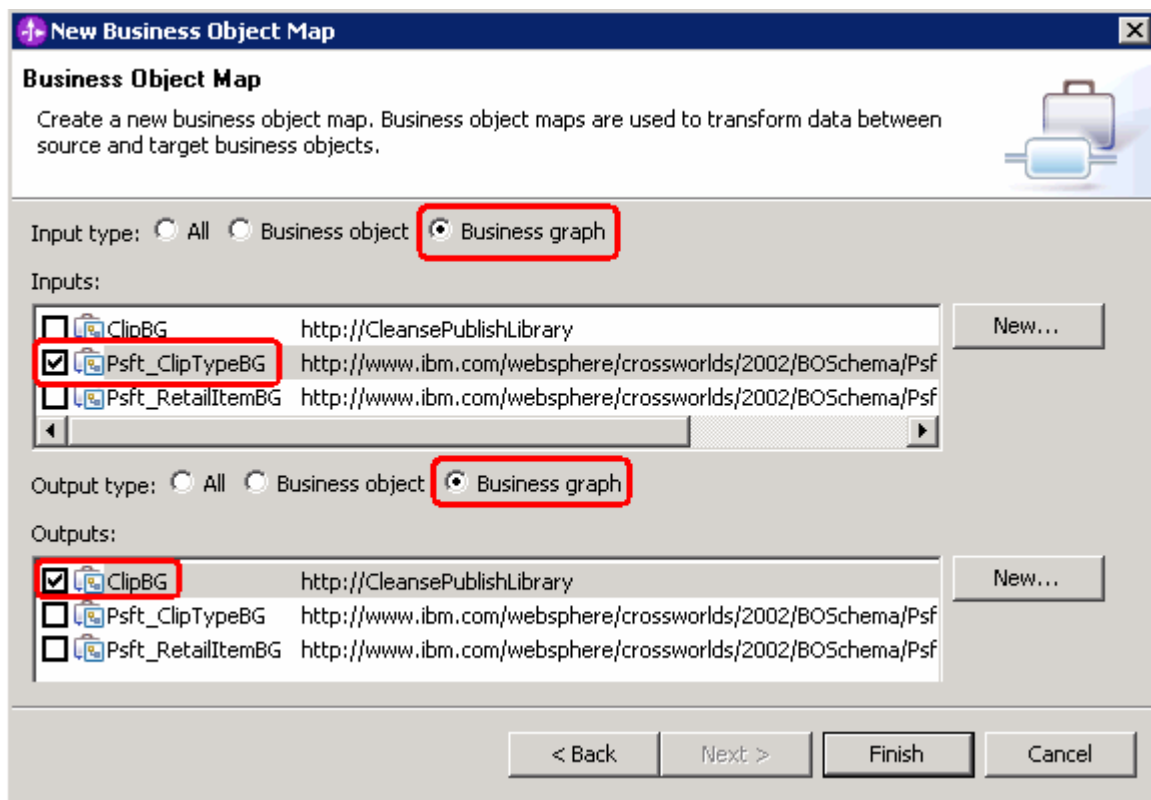
- 1) Module: **PeopleSoftAdapter**
- 2) Namespace: (accept the default)
- 3) Folder: (accept the default)
- 4) Name: **PsftClipTypeBGToClipBG**



\_\_ g. Click **Next**

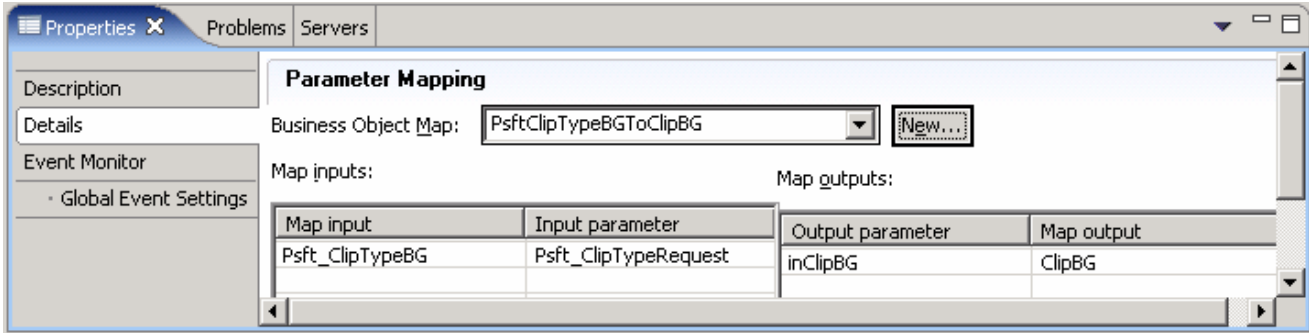
\_\_ h. In the following panel, for both the **Input Type** and **Output Type**, select the radio button next to **Business Graph** so that the lists are filtered to only show Business Graphs

- 1) Ensure the check box next to **Psft\_ClipTypeBG** in the Inputs section is selected
- 2) Ensure the check box next to **ClipBG** in the Outputs section is selected



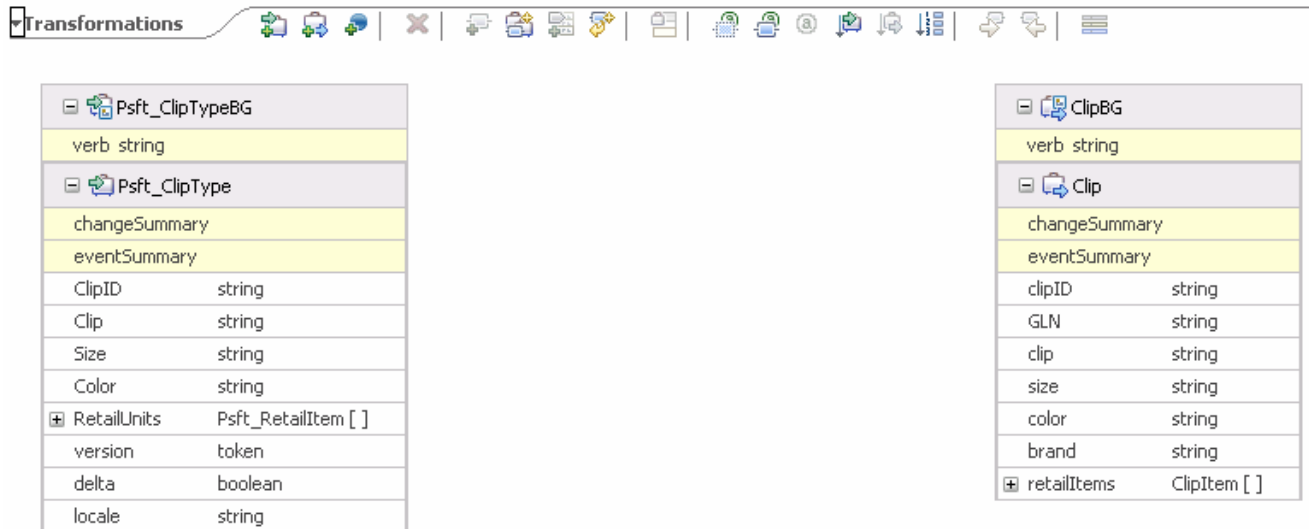
\_\_ i. Click **Finish**

- \_\_\_ j. Save the Interface Map (**Ctrl-S**)
- \_\_\_ k. Now in the properties view while the Details tab is selected, select **PsftClipTypeBGToClipBG** from the drop down list for '**Data Object Map**' as shown below:



- \_\_\_ l. Save the changes (**Ctrl + S**)

- \_\_\_ 5. Open the **PsftClipTypeBGToClipBG** map that you just created in the Map Editor
  - \_\_\_ a. In the Business Integration view, expand **PeopleSoftAdapter** → **Mapping** → **Data Maps**
  - \_\_\_ b. Double click on **PsftClipTypeBGToClipBG** to open it in the Map Editor



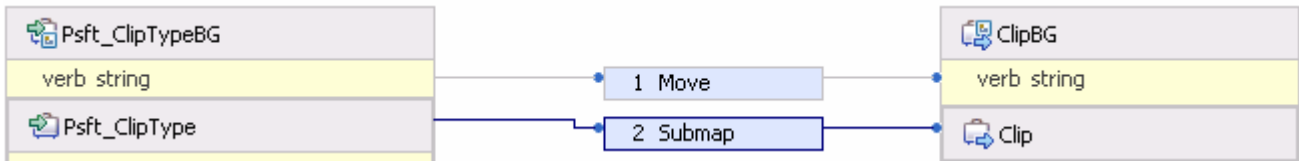
- \_\_\_ 6. Map the verb (While mapping a BG, you always want to move the verb)
  - \_\_\_ a. Click and hold on the **verb** in Psft\_ClipTypeBG and drag to the **verb** in ClipBG



- \_\_\_ b. Notice that the default transformation is move, which is what you want in this case

- \_\_\_ 7. Map the contained Business Objects

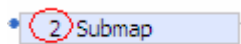
\_\_\_ a. Click and hold on **Psft\_ClipType** and drag to **Clip**



**Note:** Notice that in this case, since you are mapping from BO to BO, the default transformation is Submap

\_\_\_ 8. Create the Submap

\_\_\_ a. Click over the left of the Submap so that you can access information about the Submap in the Properties view

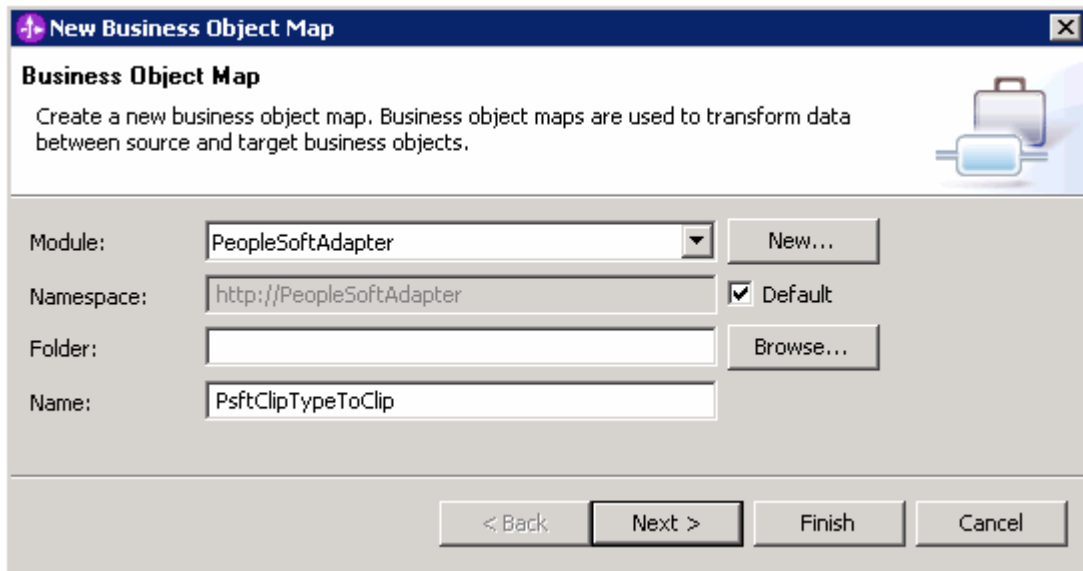


\_\_\_ b. While the left side of the Submap is selected, click on the **Details** tab in the Properties view

\_\_\_ c. Click the **New...** button to open the New Business Object Map dialog

\_\_\_ d. In the New Business Object Mapping dialog, enter following information:

- 1) Module: **PeopleSoftAdapter**
- 2) Namespace: (accept the default)
- 3) Folder: (accept the default)
- 4) Name: **PsftClipTypeToClip**

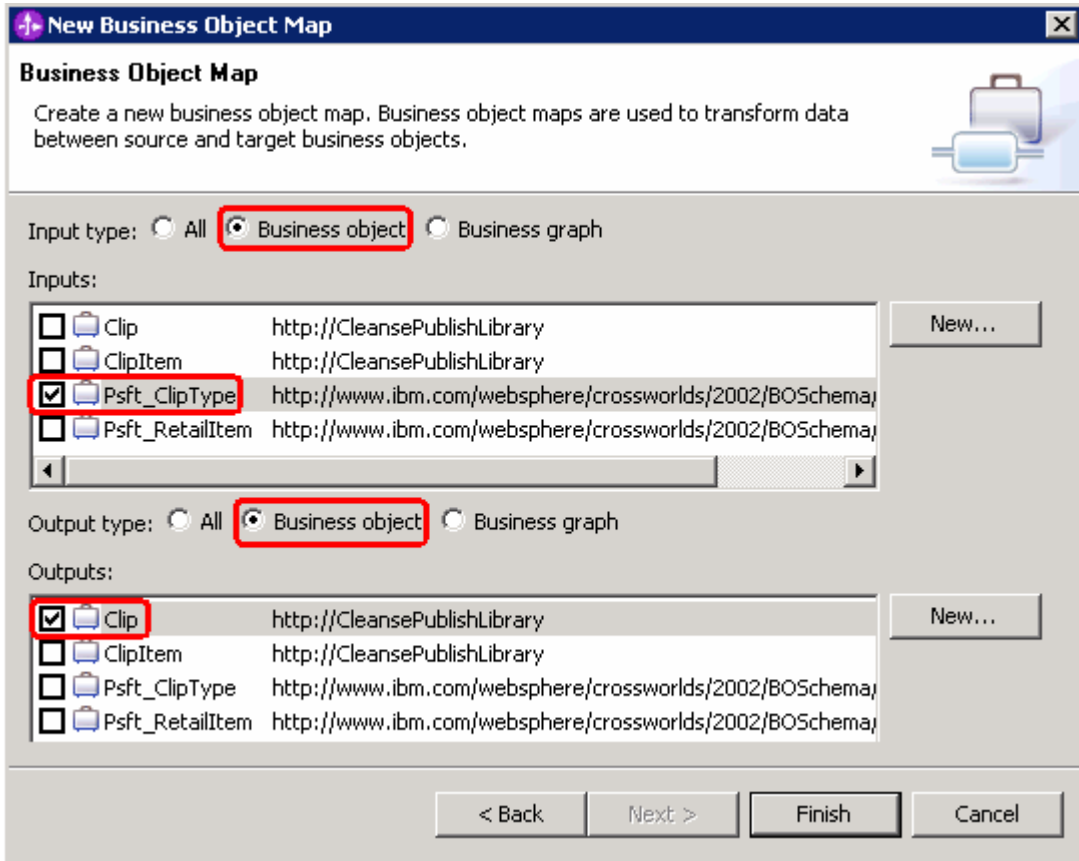


\_\_\_ e. Click **Next**

\_\_\_ f. In the following panel, for both the **Input Type** and **Output Type**, select the radio button next to **Business Object** so that the lists are filtered to only show Business Objects



- 1) For the Inputs, ensure that the check box next to **Psft\_ClipType** is selected
- 2) For the Outputs, ensure that the check box next to **Clip** is selected



\_\_\_ g. Click **Finish**. This action opens the map editor for the new Submap you just created as shown below



Psft_ClipType	
ClipID	string
Clip	string
Size	string
Color	string
⊕ RetailUnits	Psft_RetailItem [ ]
version	token
delta	boolean
locale	string

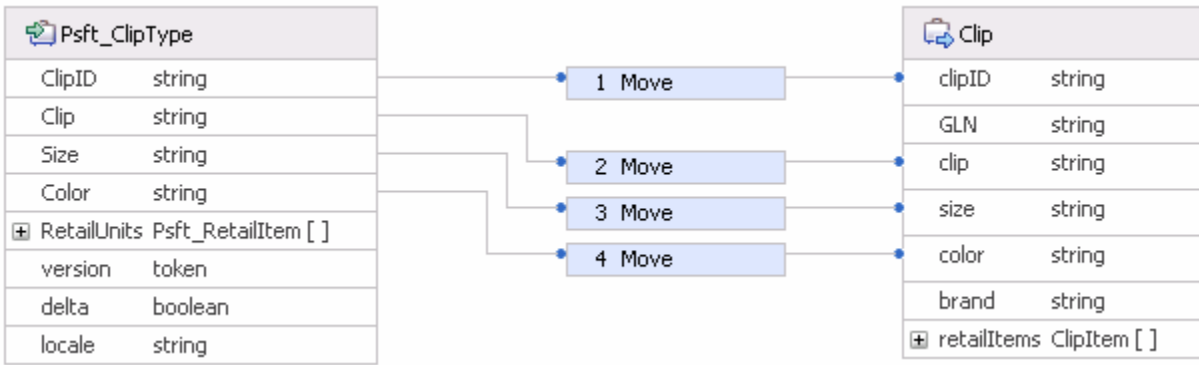
Clip	
clipID	string
GLN	string
clip	string
size	string
color	string
brand	string
⊕ retailItems	ClipItem [ ]

\_\_\_ 9. Map the fields from **Psft\_ClipType** to **Clip**

\_\_\_ a. By using the click, hold and drag technique, define move transformations for the following elements:

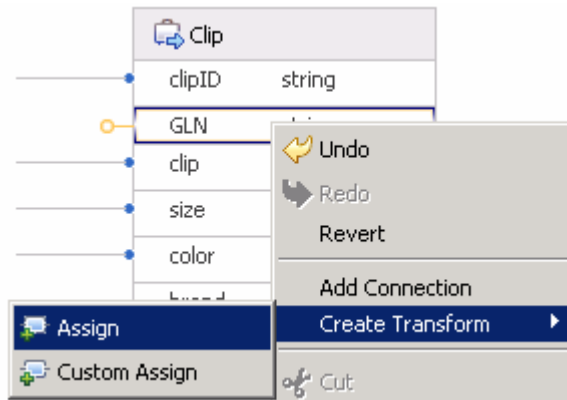
- 1) ClipID → clipID

- 2) Clip → clip
- 3) Size → size
- 4) Color → color

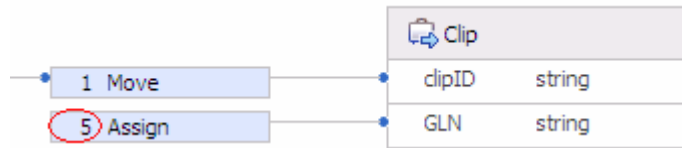


\_\_\_ b. The GLN field is used to hold the Global Location Number, a 13 digit number which uniquely identifies Clips and Tacks Clip Division. This will be set by the map using an Assign

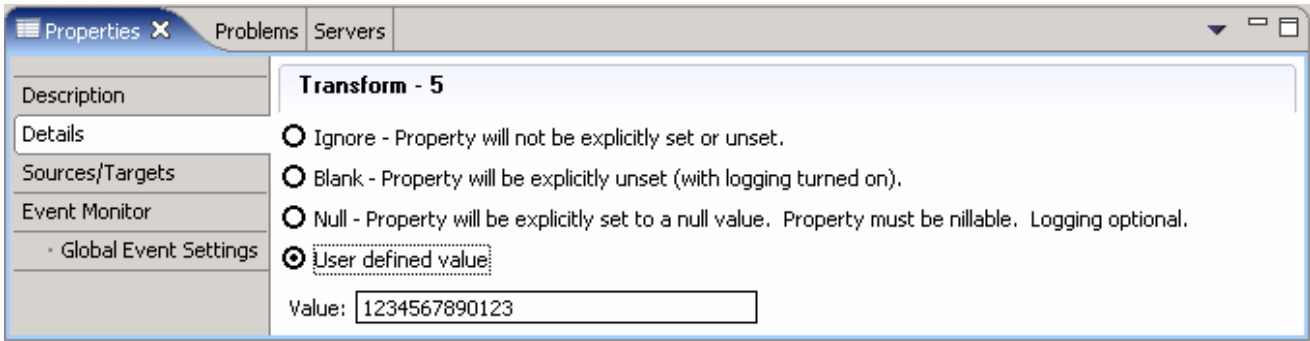
- 1) Right click on **GLN** in the Clip section and select **Create Transform → Assign** from the context menu



- 2) Click on the left of the **Assign** icon to view its properties in the Properties view



- 3) While the left side of the Assign is selected, click on the **Details** tab in the Properties view
- 4) Ensure the radio button next to **User defined value** is selected
- 5) Enter **1234567890123** in the **Value** field

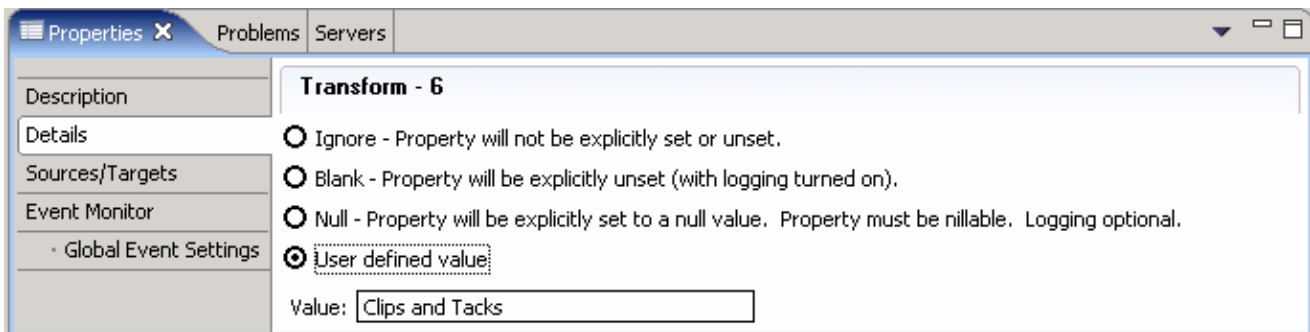


\_\_ c. The brand field is also set by the map to Clips and Tacks

- 1) Right click on **brand** in the Clip section and select **Create Transform → Assign** from the context menu
- 2) Click on the left of the **Assign** icon to view its properties in the Properties view

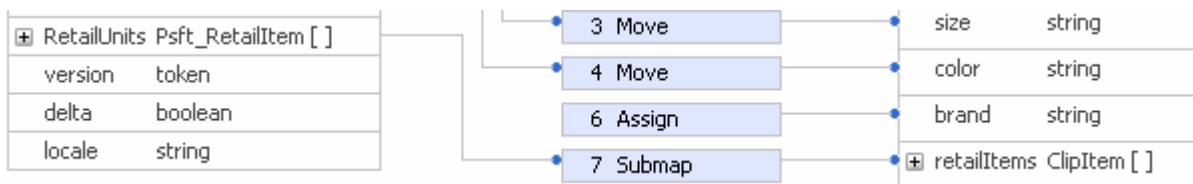


- 3) While the left side of the Assign is selected, click on the **Details** tab in the Properties view
- 4) Ensure the radio button next to **User defined value** is selected
- 5) Enter **Clips and Tacks** in the **Value** field



\_\_ d. Map **RetailUnits** to **retailItems**. This will require a Submap

- 1) Drag a line between **RetailUnits Psft\_RetailItem [ ]** and **retailItems ClipItem [ ]** to define a Submap transform



- 2) Click on the left of the **Submap** icon to view its properties in the Properties view
- 3) While the left side of the Submap is selected, click on the **Details** tab in the Properties view

- 4) Click the **New...** button to open the New Business Object Map dialog
- 5) In the New Business Object Mapping dialog, enter following information:
  - a) Module: **PeopleSoftAdapter**
  - b) Namespace: (accept the default)
  - c) Folder: (accept the default)
  - d) Name: **PsftRetailItemToClipItem**
- 6) Click **Next**
- 7) In the following panel, for both the **Input Type** and **Output Type**, select the radio button next to **Business Object** so that the lists are filtered to only show Business Objects
  - a) Ensure that the check box next to **Psft\_RetailItem** is selected in the Inputs section
  - b) Ensure that the check box next to **ClipItem** is selected in the Outputs section
- 8) Click **Finish**. This action opens the map editor for the Submap you just created

▼ Transformations

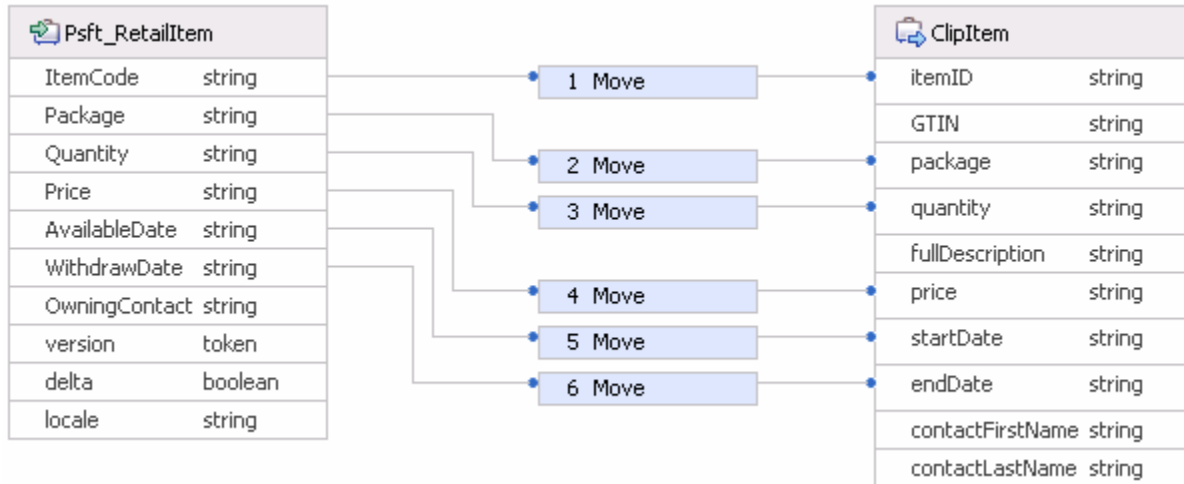
Psft_RetailItem	
ItemCode	string
Package	string
Quantity	string
Price	string
AvailableDate	string
WithdrawDate	string
OwningContact	string
version	token
delta	boolean
locale	string

ClipItem	
itemID	string
GTIN	string
package	string
quantity	string
fullDescription	string
price	string
startDate	string
endDate	string
contactFirstName	string
contactLastName	string

10. Map the fields from **Psft\_RetailItem** to **ClipItem**

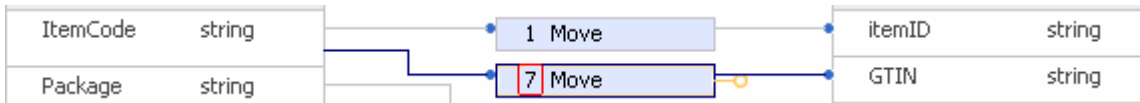
a. By using the click, hold and drag technique, define move transformations for the following elements:

- 1) ItemCode → itemID
- 2) Package → package
- 3) Quantity → quantity
- 4) Price → price
- 5) AvailableDate → startDate
- 6) WithdrawDate → endDate

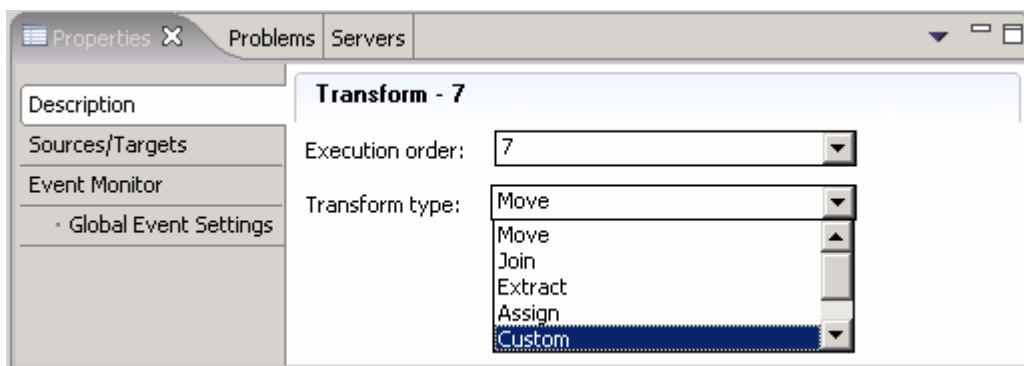


\_\_\_ b. GTIN is a Global Trade Item Number is a 14 digit number which uniquely identifies this saleable item. It is composed of a portion that is unique to ClipsAndTacks.com and a portion that is a unique identifier within ClipsAndTacks.com. This will be accomplished using a custom transform containing a Java snippet that concatenates a constant for the ClipsAndTacks.com portion and the item code for this retail item

- 1) Draw a line from **ItemCode** to **GTIN** which will create a move transform
- 2) Click on the left of the **Move** icon to view its properties in the Properties view



- 3) While the left side of the Move is selected, click on the **Description** tab in the Properties view
- 4) Use the Transform type dropdown list box to select **Custom**



- 5) Now select the **Details** tab
- 6) Select the radio button next to **Java**
- 7) Click on **Yes** for the Question dialog that pops up
- 8) With the cursor placed in the text entry box, use **Ctrl + Spacebar** to get the context assist dropdown

Visual Java

```
// The specific type of variable Psft_RetailItem_ItemCode is java.lang.String
// The specific type of variable ClipItem_GTIN is java.lang.String
```

- △ bosBOChangeSummary BOChangeSummary
- △ bosBODataObject BODataObject
- △ bosBOFactory BOFactory
- △ ClipItem\_GTIN Object
- △ execContext ExecutionContext

9) Select **ClipItem\_GTIN** from the context menu and hit **Enter**

10) Enter a **space**, **=**, a **space**, **"123456789"**, a **space**, **+**, a **space** and then click **Ctrl-Spacebar** again for context assist

Visual Java

```
// The specific type of variable Psft_RetailItem_ItemCode is java.lang.String
// The specific type of variable ClipItem_GTIN is java.lang.String
ClipItem_GTIN = "123456789" +
```

- △ bosBOChangeSummary BOChangeSummary
- △ bosBODataObject BODataObject
- △ bosBOFactory BOFactory
- △ ClipItem\_GTIN Object
- △ execContext ExecutionContext
- △ mapService MapService
- △ Psft\_RetailItem\_ItemCode Object
- △ relService RelationshipService
- △ logger Logger

11) Select **Psft\_RetailItem\_ItemCode** from the context menu and hit **Enter**

12) Enter **;** (**semi colon**) and the final Java snippet must look like in the following image:

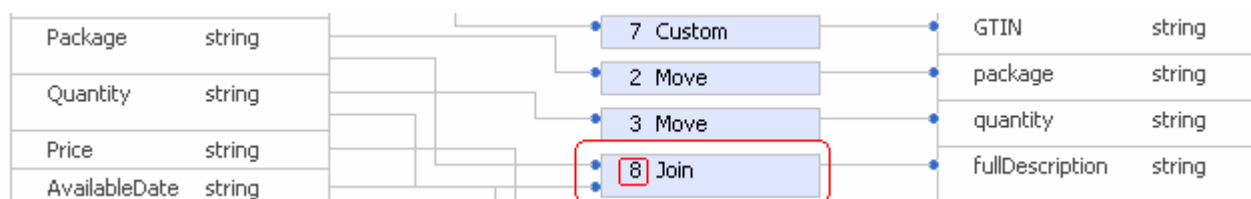
Visual Java

```
// The specific type of variable Psft_RetailItem_ItemCode is java.lang.String
// The specific type of variable ClipItem_GTIN is java.lang.String
ClipItem_GTIN = "123456789" + Psft_RetailItem_ItemCode;
```

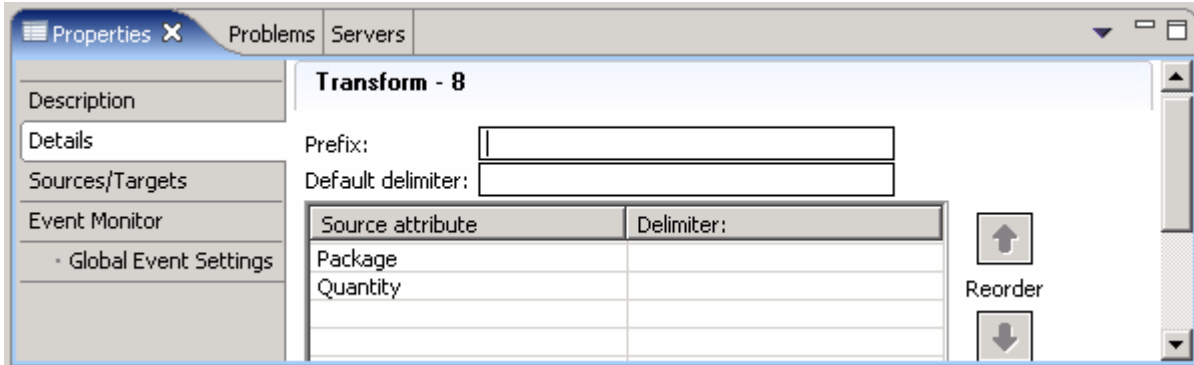
\_\_ c. Handle the fullDescription field by combining the Package and Quantity fields using a Join transform

1) Drag a line from **Package** to **fullDescription** and you will see a Move transform generated

2) Drag a line from **Quantity** to **fullDescription** and you will see the transform changed to a Join

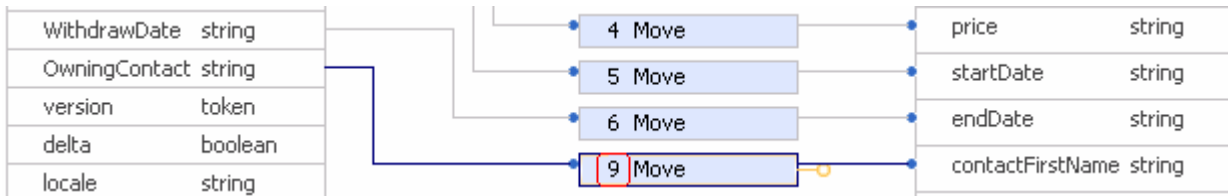


- 3) Click on the left of the **Join** to bring up this transform in the Properties view
- 4) While the left of the Join is selected, click on the **Details** tab in the **Properties** view. You will see the fields that are part of the Join. You will also notice that there is the ability to reorder the fields and to specify a delimiter to be inserted between the fields. In this case you will use a blank for the delimiter and the order is correct, so there is nothing else to do

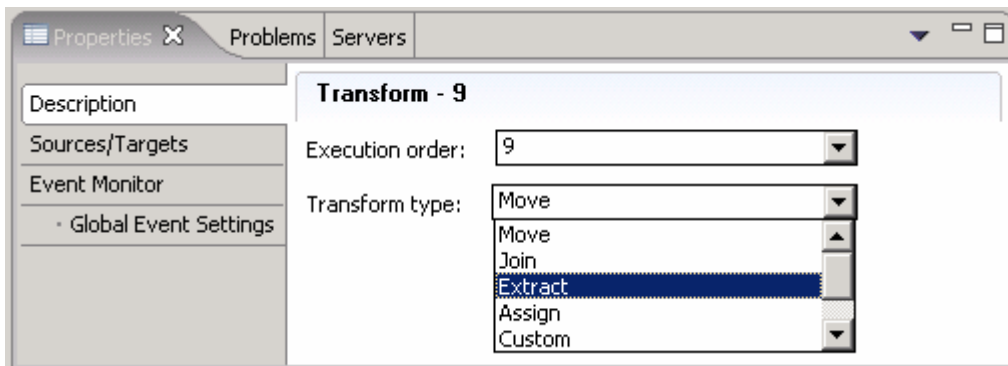


\_\_\_ d. Handle the **OwningContact** field, which is in the format of **lastName, firstName**. This field must be split between the contactFirstName and contactLastName fields which can be done using the Extract transform

- 1) Draw a line from **OwningContact** to **contactFirstName** and the default Move transform will be created

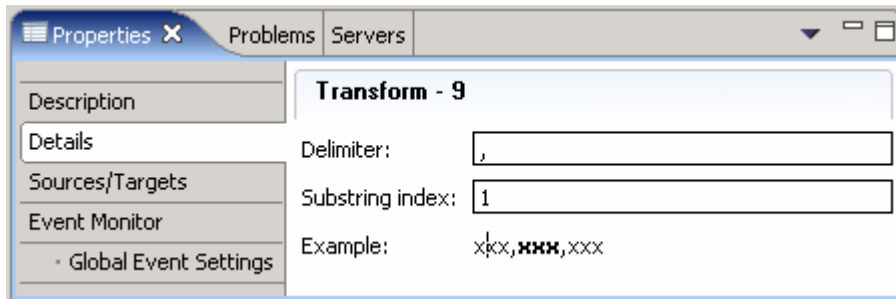


- 2) Click on the left of the **Move** to bring up the transform in the Properties view
- 3) While the left of the Move is selected, click on the **Description** tab in the **Properties** view
- 4) In the **Transform Type** field, use the dropdown box and select **Extract**

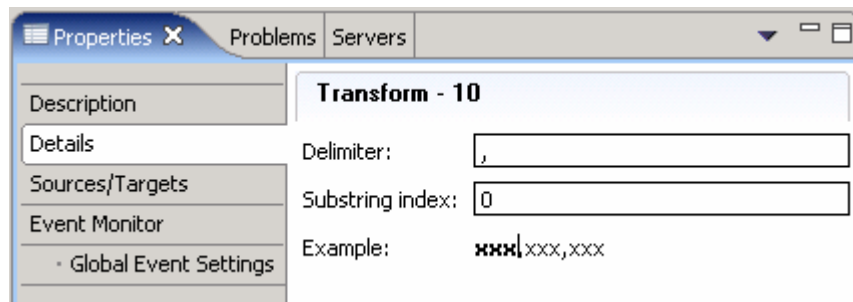


- 5) Now click to select the **Details** tab
- 6) Enter a , (comma) for the Delimiter field

- 7) Enter a **1** for the **Substring index** (this is a zero based number, so 1 is the second item) field. You will notice at the bottom there is an example which shows the delimiter and the substring index (position) you have defined

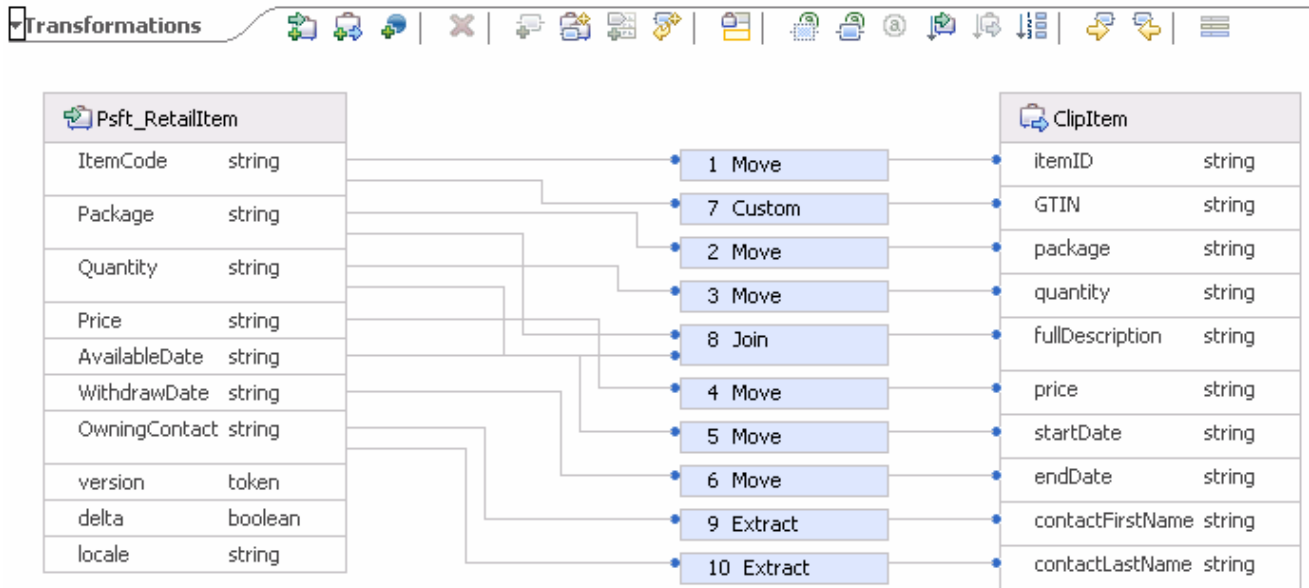


- 8) Draw a line from **OwningContact** to **contactLastName** and the default Move transform will be created
- 9) Click on the left of the **Move** to bring up the transform in the Properties view
- 10) While the left of the Move is selected, click on the **Description** tab in the **Properties** view
- 11) In the **Transform Type** field, use the dropdown box and select **Extract**
- 12) Now click to select the **Details** tab
- 13) Enter a , (comma) for the Delimiter field
- 14) Enter a **0** for the Substring index field



\_\_\_ e. The final Submap must look like as shown below:






\_\_ f. Save all the changes. Select **File → Save All** or press **Ctrl + Shift +S**

\_\_ g. **Close** all the Business Object editors

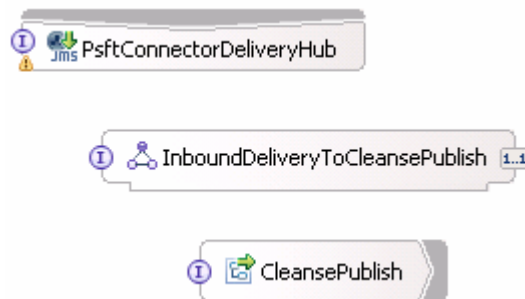
**Note:** There must not be any errors reflected in the Problems view at this time.

\_\_\_ 11. You are now ready to add the Interface Map to the wiring diagram

\_\_ a. Expand **PeopleSoftAdapter** in the Business Integration view and double click on **Assembly Diagram** (  **Assembly Diagram** ) to open it in an Assembly Diagram editor

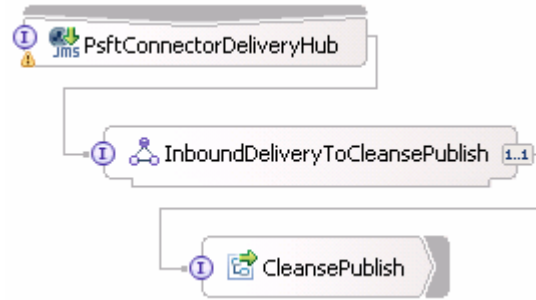
\_\_ b. In the Business Integration view, navigate to **PeopleSoftAdapter → Mapping → Interface Maps**

\_\_ c. Select **InboundDeliveryToCleansePublish** and drag it onto the Assembly Editor



\_\_ d. Draw a wire from the **PsftConnectorDeliveryHub** Export to the **InboundDeliveryToCleansePublish** Interface Map


\_\_ e. Similarly draw a wire from the **InboundDeliveryToCleansePublish** Interface Map to the **CleansePublish** Import



\_\_\_ f. Press **Ctrl-S** to save the Assembly Diagram


## Part 3: Define interface maps and data maps for Siebel

In this part you will create the Interface Map needed to map the Publish interface of the Export which is called by the CleansePublish module to the OutboundRequest interface supported but the Import used to talk to the Siebel Adapter. Within the Interface Map you will define the Data Maps required to map the GBOs to the Siebel ASBOs.

\_\_\_ 1. Expand **SiebelAdapter** in the Business Integration view and double click on **Assembly Diagram** ( **Assembly Diagram**) to open it in an Assembly Diagram editor


\_\_\_ 2. Add the **Export** that will be used to call this module from the CleansePublish module

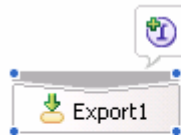
\_\_\_ a. Drag an **Export** onto the Assembly diagram

- 1) Click on **Export** icon from Assembly Diagram tray ()
- 2) Click or drag import to the Assembly diagram



\_\_\_ b. Hover over Export or click on the Export to make the add interface icon appear

- 1) Click "I" icon () to add an interface

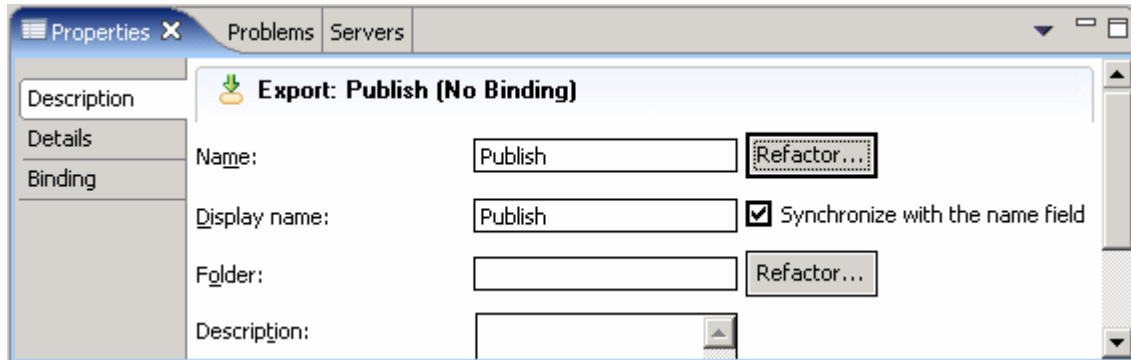


- 2) Select **Publish** from the Add Interface dialog and click **OK**

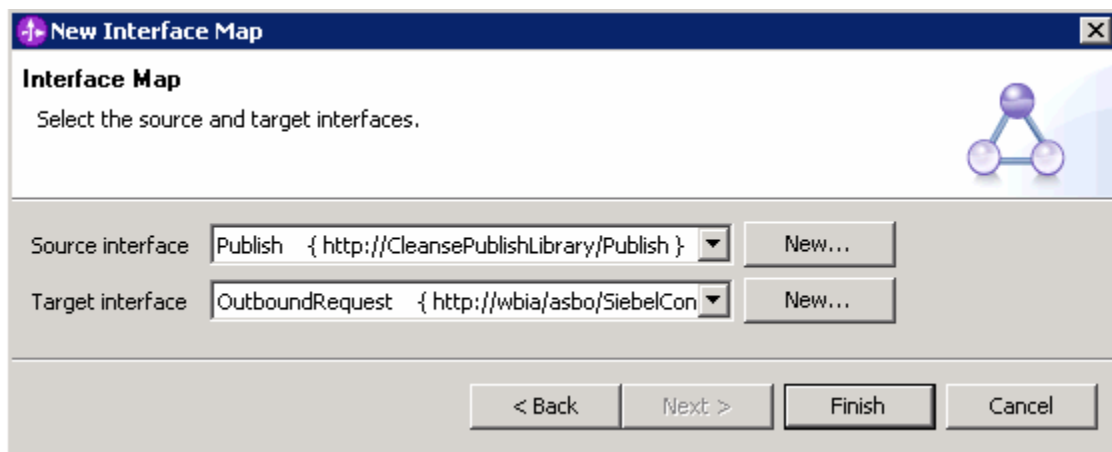
\_\_\_ c. Refactor the Description for the Export, **Export1**

- 1) Select the Export, **Export1** on the Assembly Diagram
- 2) In the Properties view select the **Description** tab
- 3) Rename the **Display name** and **Name** to **CleansePublish**

**Hint:** Modifying Name field automatically synchronizes with the Display name with the name filed when the check box next to “Synchronize with the name filed” is selected.



- \_\_\_ d. Press **Ctrl-S** to save the Assembly Diagram
- \_\_\_ 3. Add an Interface Map that will map the Publish interface to the OutboundRequest interface
  - \_\_\_ a. In the Business Integration view, expand **SiebelAdapter** → **Mapping**
  - \_\_\_ b. Right click on **Interface Maps** and select **New** → **Interface Map** from the context menu
  - \_\_\_ c. In the **New Interface Map** dialog, enter the following values:
    - 1) Module: **SiebelAdapter**
    - 2) Namespace: (accept the default)
    - 3) Folder: (accept the default)
    - 4) Name: **PublishToOutboundRequest**
  - \_\_\_ d. Click **Next**
  - \_\_\_ e. In the following panel, use the drop down list boxes to select **Publish** as the Source interface and **OutboundRequest** as the Target interface



- \_\_\_ f. Click **Finish** and this action opens the Interface Mapping editor with the Operation Mappings pane visible

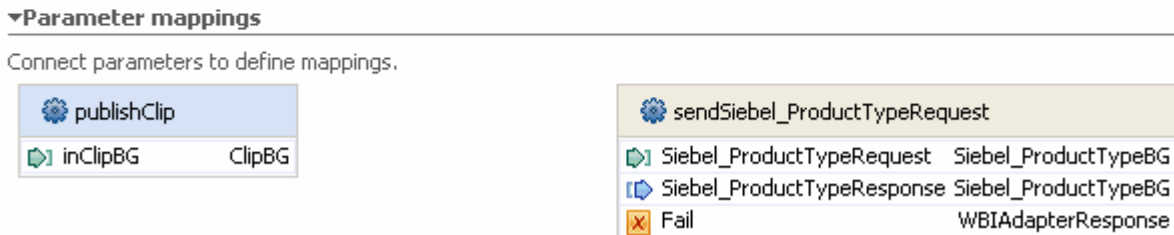
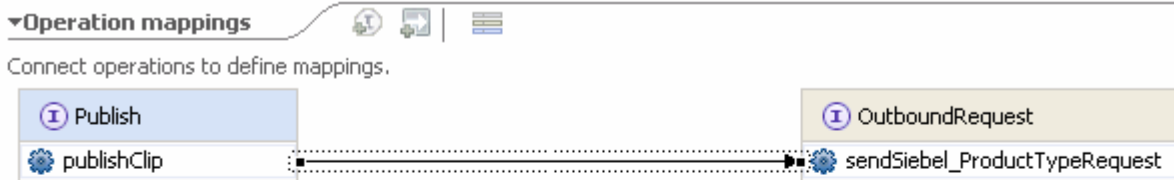


4. Define the PublishToOutboundRequest Interface Map

a. Draw a line from the **publishClip** operation to the **sendSiebel\_ProductTypeRequest** operation



b. Click on the line (wire) between the two operations, and the Parameter mappings pane will become visible as shown below:

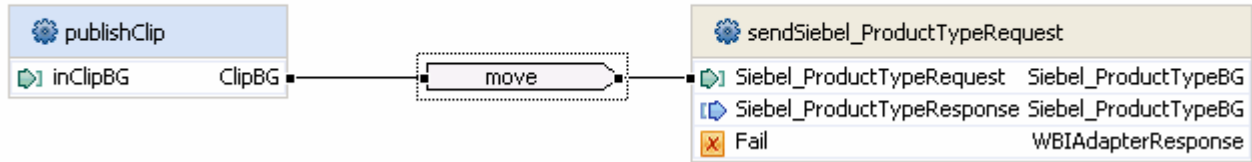


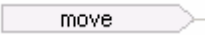
**Note:** You will notice that the **publishClip** operation only has an input while the **sendSiebel\_ProductTypeRequest** not only has an input but also has an output and a fault. In this lab you will just map the inputs and ignore the output and the fault. In the subsequent lab on relationships changes will be made to handle the output.

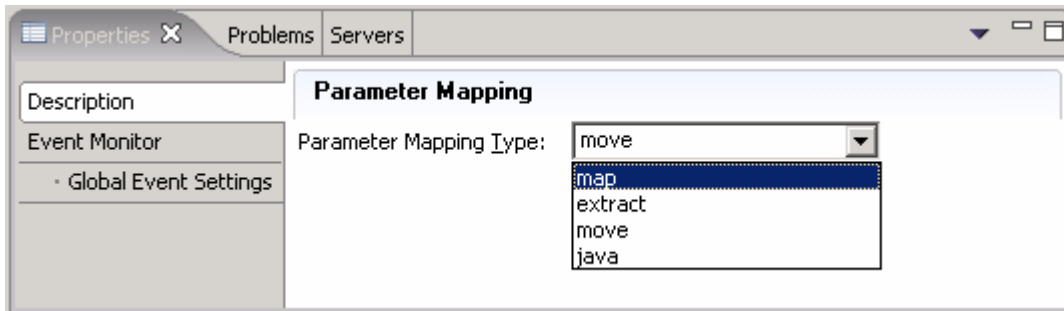
c. In the Parameter mappings section, draw a line from the **inClipBG** parameter to the **Siebel\_ProductTypeRequest** parameter. Notice that the default parameter mapping operation is a **move**. Since these parameters are of different types, you will need to change this to a map in the next step

▼Parameter mappings

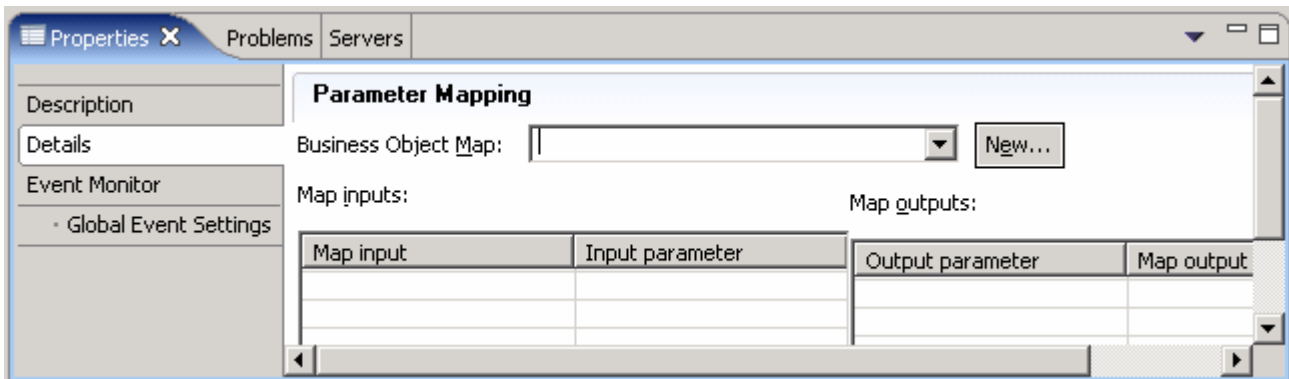
Connect parameters to define mappings.



- \_\_\_ 5. Create the Data Map to **map** the inClipBG parameter (which is a ClipBG) to the Siebel\_ProductTypeRequest parameter (which is a Siebel\_ProductTypeBG)
  - \_\_\_ a. Click on the wire (  ) to select it
  - \_\_\_ b. While the wire is selected, select the Description tab of the Properties view
  - \_\_\_ c. Use the drop down box to change the Parameter Mapping Type from move to **map**

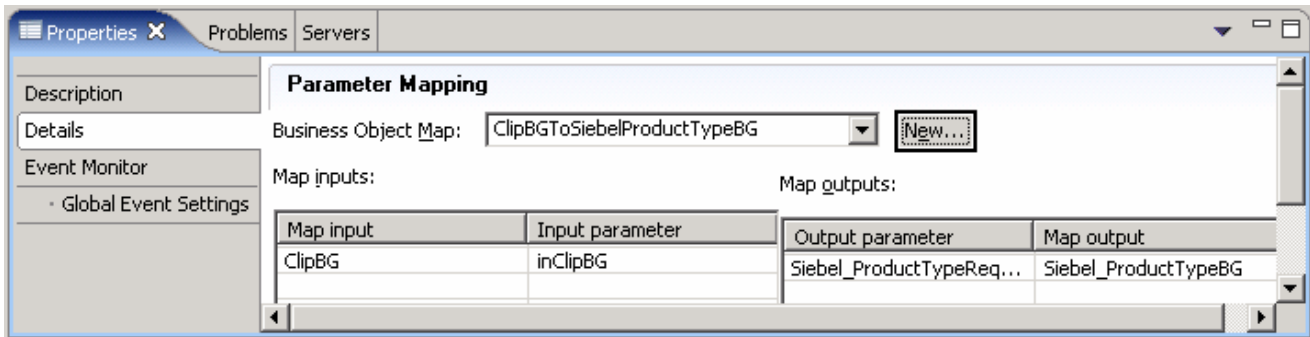


- \_\_\_ d. Now click on the **Details** tab in the Properties view

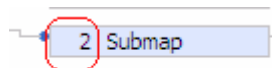


- \_\_\_ e. Click the **New...** button to open the New Business Object Mapping dialog
- \_\_\_ f. In the New Business Object Mapping dialog, enter following information:
  - 1) Module: **SiebelAdapter**
  - 2) Namespace: (accept the default)
  - 3) Folder: (accept the default)
  - 4) Name: **ClipBGToSiebelProductTypeBG**
- \_\_\_ g. Click **Next**

- \_\_\_ h. In the following panel, for both the **Input Type** and **Output Type**, select the radio button next to **Business Graph** so that the lists are filtered to only show Business Graphs
  - 1) Ensure the check box next to **ClipBG** is selected in the Inputs section
  - 2) Ensure the check box next to **Siebel\_ProductTypeBG** is selected in the Outputs section
- \_\_\_ i. Click **Finish**
- \_\_\_ j. Press **Ctrl-S** to save the Interface Map
- \_\_\_ k. Now in the properties view while the Details tab is selected, select **ClipBGToSiebelProductTypeBG** from the drop down list for 'Data Object Map' as shown below:



- \_\_\_ l. Press **Ctrl-S** to save the changes
- \_\_\_ 6. Open the **ClipBGToSiebelProductTypeBG** map that you just created in the Map Editor
  - \_\_\_ a. In the Business Integration view, expand **SiebelAdapter → Mapping → Data Maps**
  - \_\_\_ b. Double click on **ClipBGToSiebelProductTypeBG** to open it in the Map Editor
- \_\_\_ 7. Map the verb
  - \_\_\_ a. Click and hold on the **verb** in **ClipBG** and drag to the **verb** in **Siebel\_ProductTypeBG** which will create a **Move** transform
- \_\_\_ 8. Map the contained Business Objects
  - \_\_\_ a. Click and hold on **Clip** and drag to **Siebel\_ProductType** which will create a **Submap** transform
- \_\_\_ 9. Create the Submap
  - \_\_\_ a. Click on the left side of the **Submap** so that you can access properties of the **Submap** in the **Properties** view



- \_\_\_ b. While the left of the Submap is selected, click to select the **Details** tab in the **Properties** view
- \_\_\_ c. Click the **New...** button to open the **New Business Object Map** dialog
- \_\_\_ d. Enter the following values in the **New Business Object Map** dialog

- 1) Module: **SiebelAdapter**
- 2) Namespace: (accept the default)
- 3) Folder: (accept the default)
- 4) Name: **ClipToSiebelProductType**

\_\_ e. Click **Next**

\_\_ f. In the following panel, for both the **Input Type** and **Output Type**, select the radio button next to **Business Object** so that the lists are filtered to only show Business Objects

- 1) Ensure the check box next to **Clip** is selected in the **Inputs** section
- 2) Ensure the check box next to **Siebel\_ProductType** is selected in the **Outputs** section

\_\_ g. Click **Finish**. This action opens the map editor for the map you've just created as shown below:

▼ Transformations

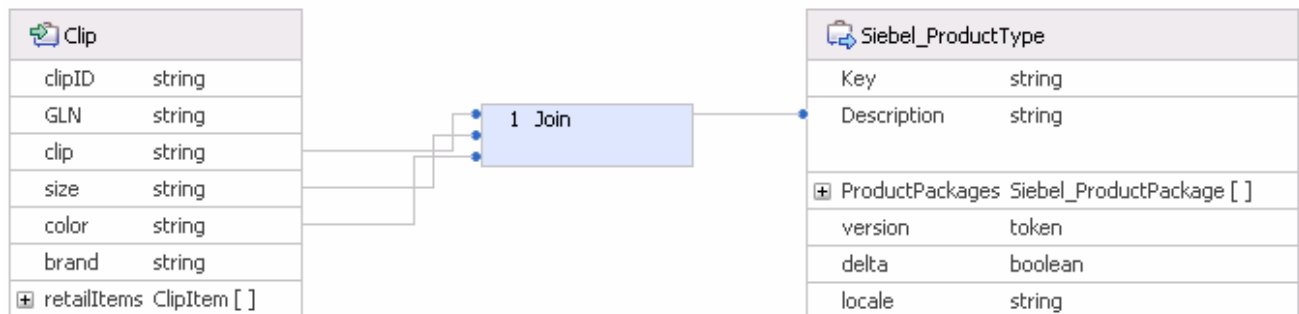
Clip	
clipID	string
GLN	string
clip	string
size	string
color	string
brand	string
+	retailItems ClipItem [ ]

Siebel_ProductType	
Key	string
Description	string
+	ProductPackages Siebel_ProductPackage [ ]
version	token
delta	boolean
locale	string

\_\_\_ 10. Map the fields from **Clip** to **Siebel\_ProductType**

\_\_ a. The first transform will be to use a Join to initialize the Description field with the clip, size and color fields

- 1) Draw a line from **clip** to **Description**. This will result in the default Move transformation
- 2) Draw a line from **size** to **Description**. This will change the transformation to a Join
- 3) Draw a line from **color** to **Description**. Your map should now look like as shown below:



\_\_ b. Map **retailItems** to **ProductPackages**. This requires a Submap



- 1) Drag a line between **ClipItem [ ]** and **Siebel\_ProductPackage [ ]** to define a **Submap** transform



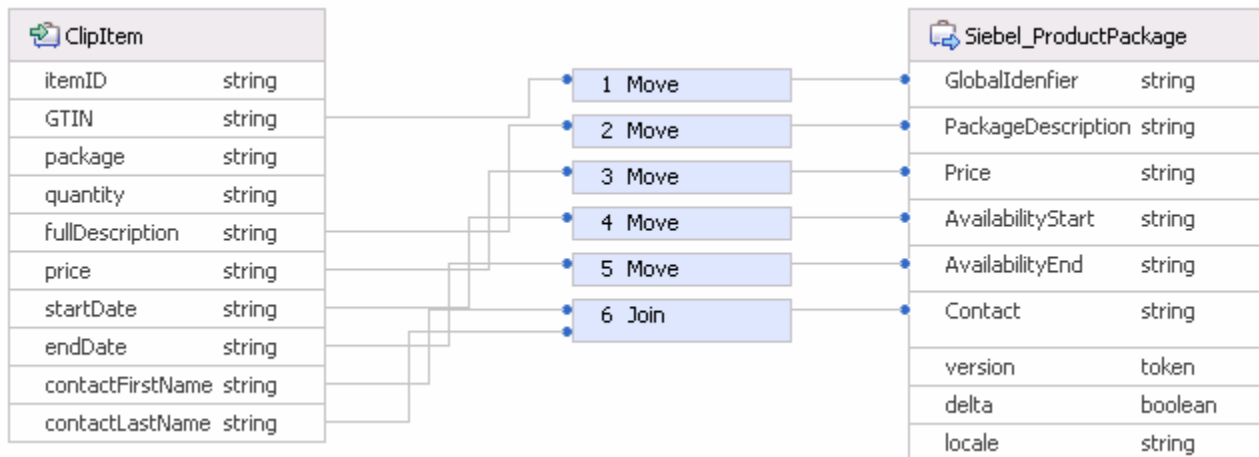
- 2) Click on the left of the **Submap** to access its properties in the Properties view
- 3) While the left of the Submap is selected, click on the **Details** tab in the Properties view
- 4) Click the **New...** button to open the **New Business Object Map** dialog
- 5) Enter the following values in the **New Business Object Map** dialog
  - a) Module: **SiebelAdapter**
  - b) Namespace: (accept the default)
  - c) Folder: (accept the default)
  - d) Name: **ClipItemToSiebelProductPackage**
- 6) Click **Next**
- 7) In the following panel, for both the **Input Type** and **Output Type**, select the check box next to **Business Object** so that the lists are filtered to only show Business Objects
  - a) Ensure the check box next to **ClipItem** is selected in the **Inputs** section
  - b) Ensure the check box next to **Siebel\_ProductPackage** is selected in the **Outputs** section
- 8) Click **Finish**. This action will open the map editor for the map you just created



ClipItem	
itemID	string
GTIN	string
package	string
quantity	string
fullDescription	string
price	string
startDate	string
endDate	string
contactFirstName	string
contactLastName	string

Siebel_ProductPackage	
GlobalIdentifier	string
PackageDescription	string
Price	string
AvailabilityStart	string
AvailabilityEnd	string
Contact	string
version	token
delta	boolean
locale	string

11. Map the fields from ClipItem to Siebel\_ProductPackage
- a. By using the click, hold and drag technique, define **Move** transformations for the following
    - 1) GTIN → GlobalIdentifier
    - 2) fullDescription → PackageDescription
    - 3) price → Price
    - 4) startDate → AvailabilityStart
    - 5) endDate → AvailabilityEnd
  - b. The Contact field will contain the name in the firstName, lastName format. Create a **Join** transformation to do this
    - 1) Draw a line from **contactFirstName** to **Contact**
    - 2) Draw a line from **contactLastName** to **Contact**
  - c. Your map should look like as shown below:




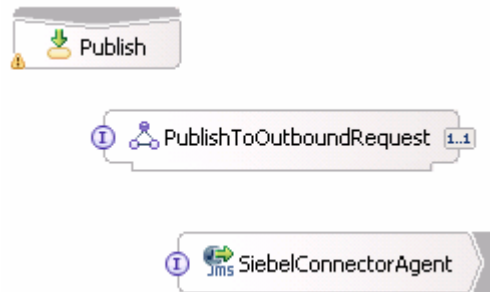
- \_\_\_ d. Save all the changes. Select **File → Save All** or press **Ctrl + Shift +S**
- \_\_\_ e. **Close** all the Business Object editors

---

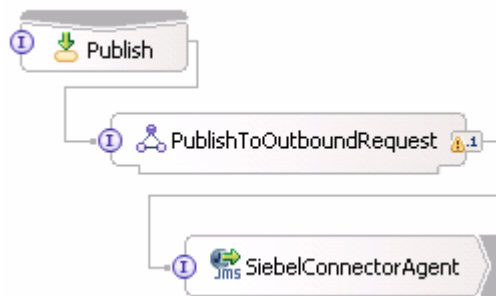
**Note:** There must not be any errors reflected in the Problems view at this time.

---

- \_\_\_ 12. You are now ready to add the Interface Map to the wiring diagram
  - \_\_\_ a. Expand **SiebelAdapter** in the Business Integration view and double click on **Assembly Diagram** ( **Assembly Diagram**) to open it in an Assembly Diagram editor
  - \_\_\_ b. In the Business Integration view, expand **SiebelAdapter → Mapping → Interface Maps**
  - \_\_\_ c. Click on **PublishToOutboundRequest** and drag it onto the Assembly Editor
  - \_\_\_ d. Rearrange the assembly diagram by dragging the elements so that they look like as shown below:



- \_\_\_ 13. You are now ready to add the Interface Map to the wiring diagram
  - \_\_\_ a. Draw a wire from the **Publish** Export to the **PublishToOutboundRequest** Interface Map
  - \_\_\_ b. Draw a wire from the **PublishToOutboundRequest** Interface Map to the **SiebelConnectorAgent** Import

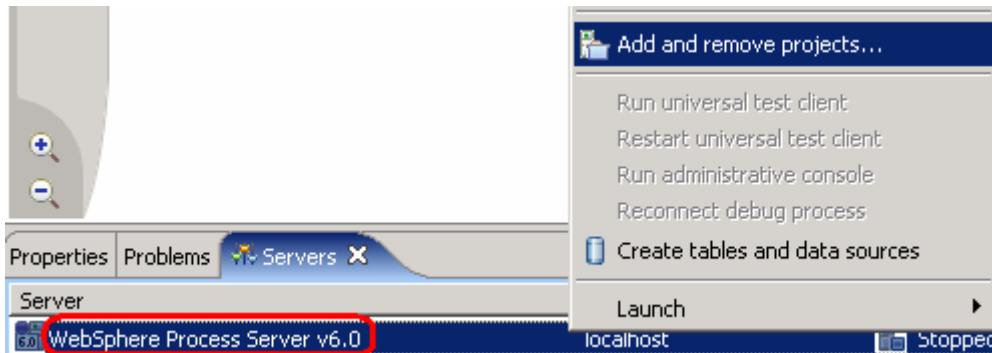


- \_\_\_ c. Press **Ctrl-S** to save the Assembly Diagram

## Part 4: Test the PeopleSoft maps

There are two ways that the maps created for PeopleSoft can be tested. The first approach would be to use the Test Component to drive the Interface Map on the input side and to see the result on the output side. This would be a unit test of just the maps. The other approach would be to use the Visual Test Connector to drive input through the Export which is wired to the Interface Map and then to use the Test Component to see the result on the output side of the Interface map. This not only tests the maps but also the wiring to ensure your module works as a whole. This is the approach you will take in this section.

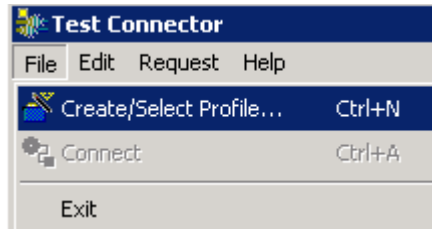
- \_\_\_ 1. Add the PeopleSoftAdapter module to the WebSphere Process Server test environment
  - \_\_\_ a. From the Server view, right click on the server and select **Add and Remove Projects...** from the popup menu.



- \_\_\_ b. In the Add and Remove Projects dialog, select the **PeopleSoftAdapterApp** project from the Available projects panel
- \_\_\_ c. Click **Add >** to add it to the Configured projects panel
- \_\_\_ d. Click **Finish**. If using a local testing environment, the server automatically starts if the it is not started at this time

If using a remote testing environment, follow the directions provided in [Task: Adding Remote Server to WebSphere Integration Developer Test Environment](#) (at the end of this document)

- \_\_\_ 2. Connect the Visual Test Connector (from now on VTC) to the WebSphere Process Server test environment. Using the connector configuration at **<LAB\_FILES>\eXchange\PeopleSoft\ASBOs\PsfConnectorWPS.cfg** follow the instructions below to use the Visual Test Connector:
  - \_\_\_ a. Ensure the WebSphere Process Server is active (the server is the JMS Broker). The VTC requires an active JMS Broker to start its connection
  - \_\_\_ b. Start the Visual Test Connector
  - \_\_\_ c. From the **start** menu, select **Programs → IBM WebSphere Business Integration Adapters → IBM WebSphere Business Integration Toolset → Visual Test Connector**. This opens a command window in the background where the process runs and the Visual Test Connector Interface in the foreground. Do not close the command window as this will cause the VTC to exit
  - \_\_\_ d. From the main menu of the VTC interface, select **Create/Select a Profile**. The Connector Profile panel opens

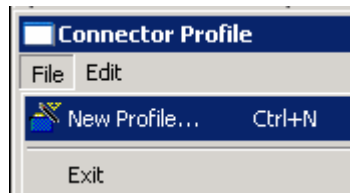


\_\_ e. Create a new PeopleSoft profile by selecting **File → New Profile...** from the main menu of the Connector Profile panel

---

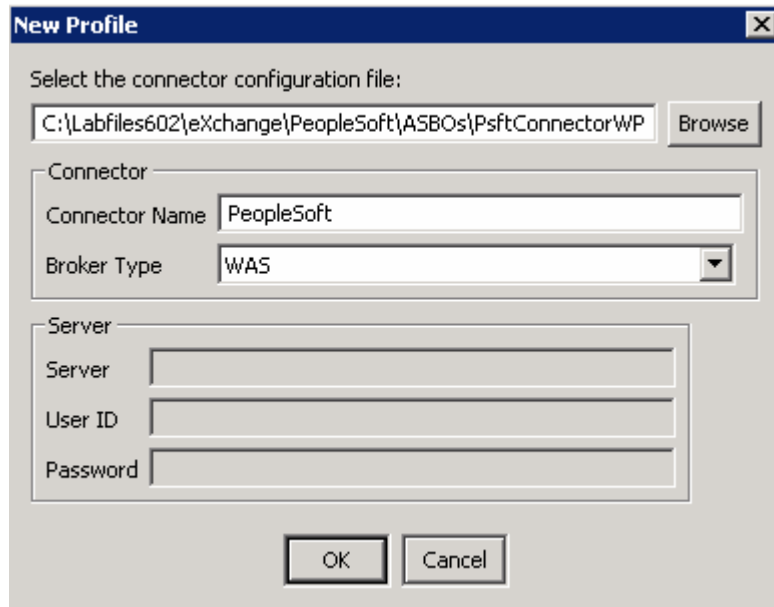
**Note:** You can skip these steps if a PeopleSoft profile has already been created and is listed.

---



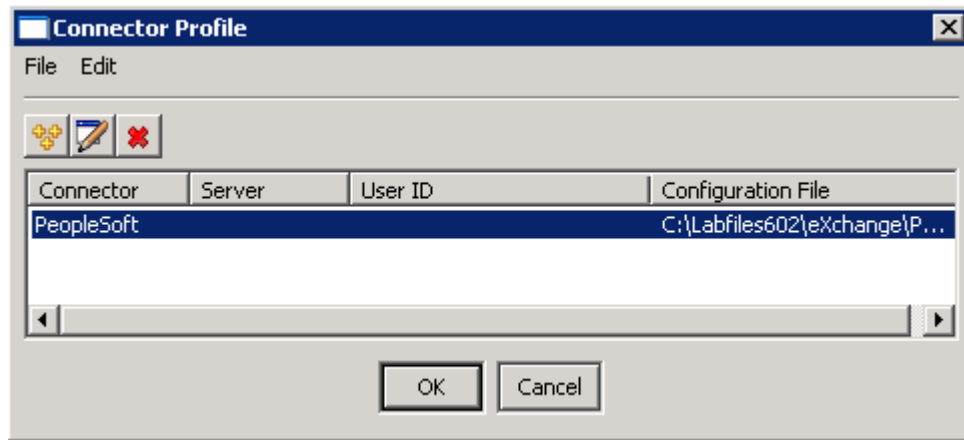
\_\_ f. Enter the following information in the New Profile panel:

- 1) Click the **Browse** button for “Select the connector configuration file:” field and select **<LAB\_FILES>\eXchange\PeopleSoft\ASBOs\PsftConnectorWPS.cfg** file
- 2) Connector Name : **PeopleSoft**
- 3) Broker Type : **WAS**

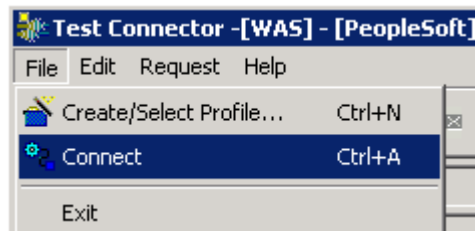


\_\_ a. Click **OK**

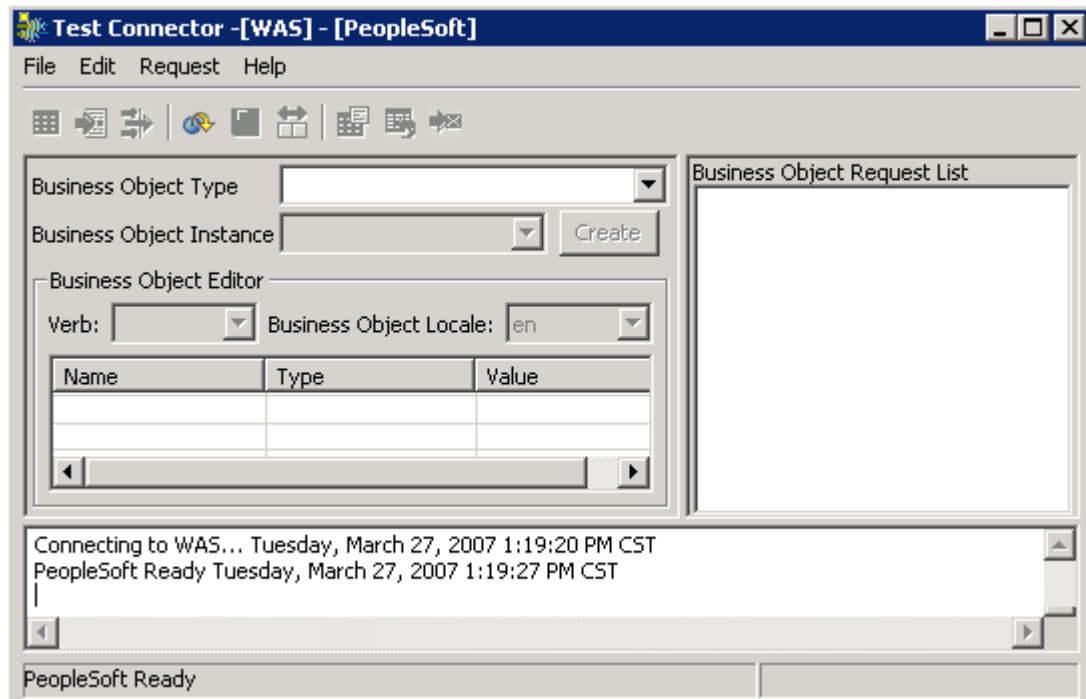
\_\_ g. Select the PeopleSoft profile you had created as shown below:



\_\_\_ h. Now in the Test Connector interface, select **File** → **Connect** from the main menu

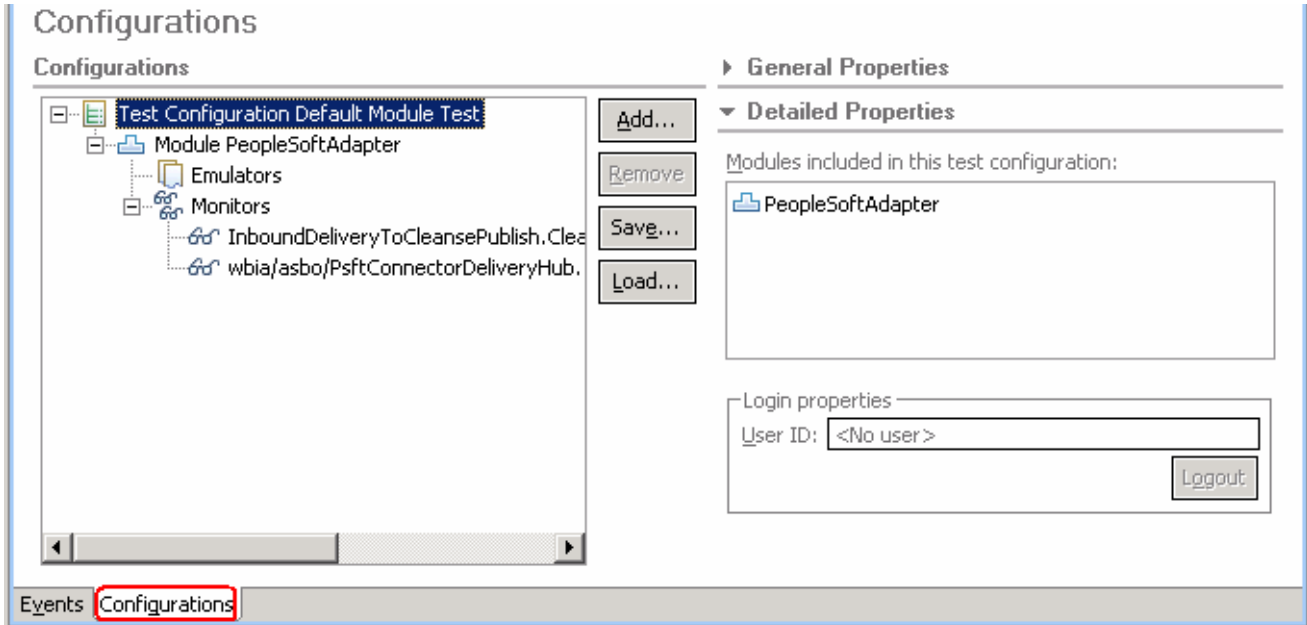


\_\_\_ i. Ensure that a successful connection is made. You can confirm this by watching the trace messages displayed at the bottom frame of the Test Connector interface as shown below:

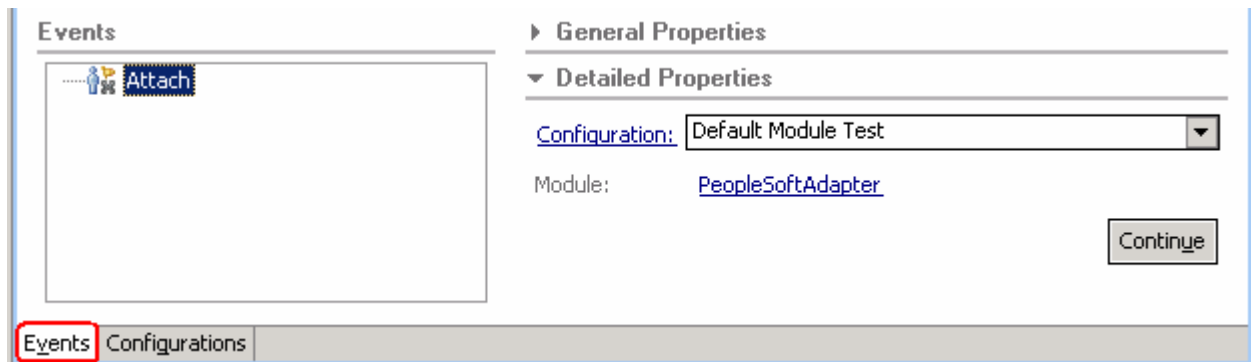


\_\_\_ 3. In the WebSphere Integration Developer, attach the test component to the PeopleSoftAdapter module

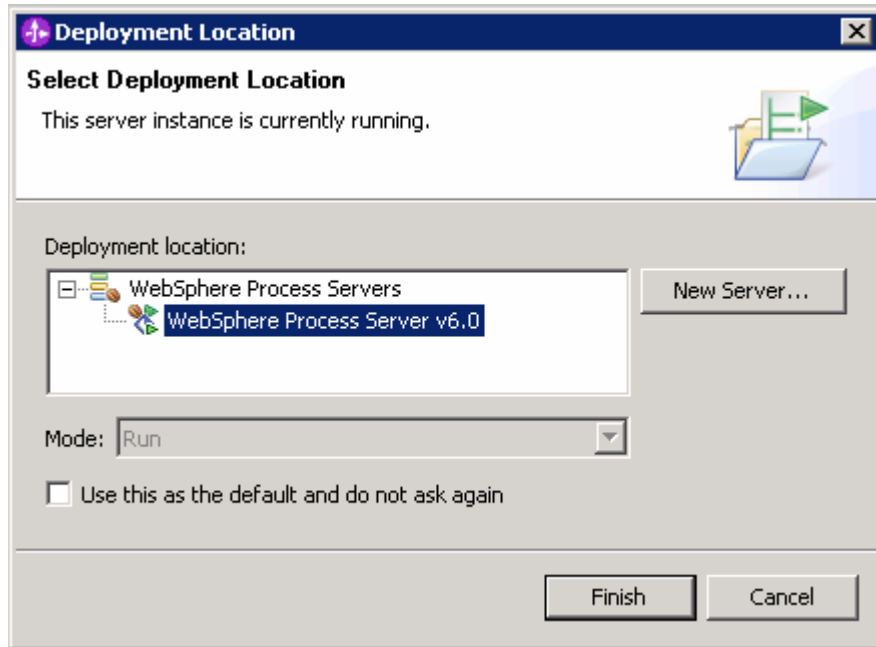
- \_\_\_ a. In the Business Integration view, right click on the **PeopleSoftAdapter** and select **Test → Attach** from the context menu. The PeopleSoftAdapter Test Component editor opens
- \_\_\_ b. Click on the **Configurations** tab at the bottom of the **Testing Module: PeopleSoftAdapter** view. You should see something like in the following image:



- \_\_\_ c. Click on the **Events** tab at the bottom of the **PeopleSoftAdapter\_Test** view. You should see something like the following screen capture. This is where the results will be shown when the application is invoked



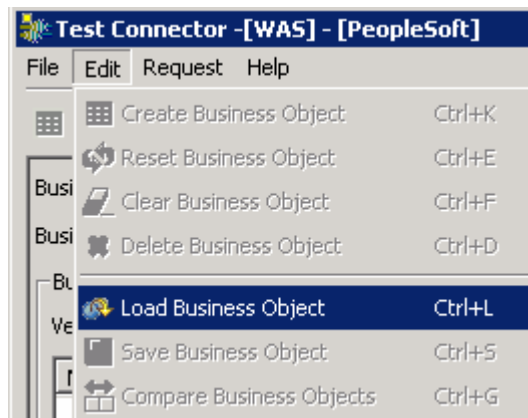
- \_\_\_ d. Click **Continue**
- \_\_\_ e. Select the **WebSphere Process Server V6.0** server in the **Choose a deployment location** dialog



\_\_\_ f. Click **Finish**. The Test Component is now ready to monitor the execution flow

\_\_\_ 4. Load a predefined business object into the Visual Test Connector

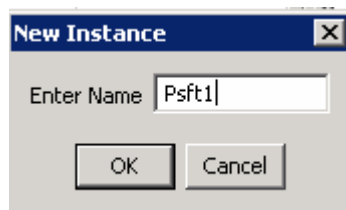
\_\_\_ a. In the main menu of the Visual Test Connector interface, select **Edit → Load Business Object**



\_\_\_ b. In the Open dialog, navigate to **<LAB\_FILES>\eXchange\PeopleSoft\ASBOInstances** and select **PsftASBO1.bo**

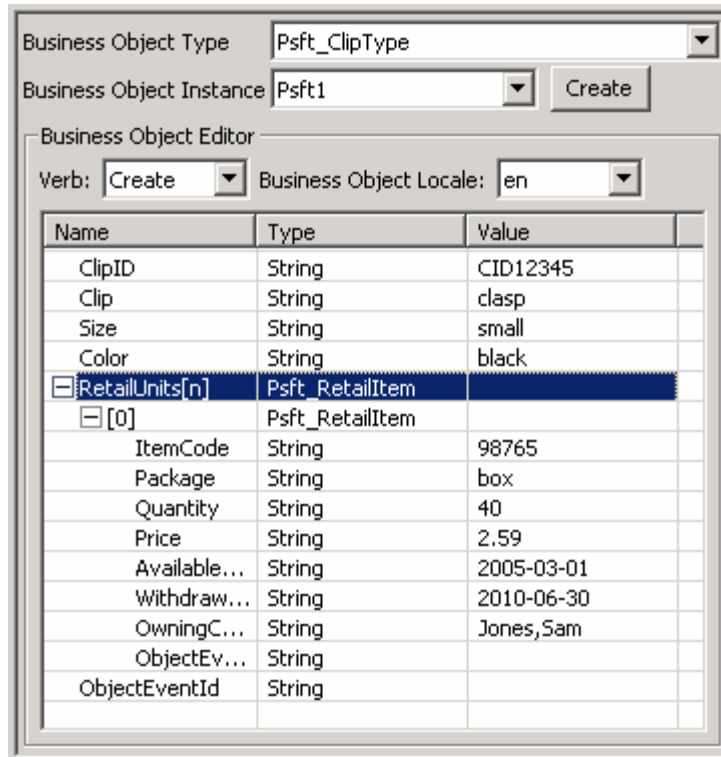
\_\_\_ c. Click **Open**

\_\_\_ d. In the New Instance dialog, enter the name to be assigned to this instance as **Psft1**





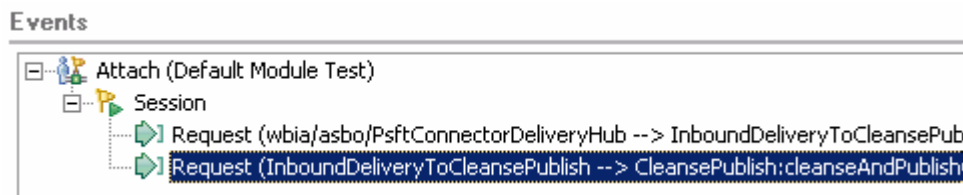
- \_\_\_ 5. Review the data in the business object
  - \_\_\_ a. Review the values assigned to the Business Object, **Psft\_ClipType**
  - \_\_\_ b. Expand **RetailUnits[n]** to expand the **Psft\_RetailItem** array
  - \_\_\_ c. Expand **[0]** to view the instance of **Psft\_RetailItem**



- \_\_\_ 6. Send the business object to the server
  - \_\_\_ a. From the main menu of the Visual Test Connector, select **Request → Send**



- \_\_\_ 7. Review the resultant business object in the server test component
  - \_\_\_ a. In the Events panel, you should see the Request Monitor for the wire between the InboundDeliveryToCleansePublish Interface Map and the CleansePublish Import



- \_\_\_ b. Click on **Request Monitor (InboundDeliveryToCleansePublish -> CleansePublish)**. The object that went over this wire will be visible on the right side in the Request parameters box

Input parameters

Name	Type	Value
[-] Psft_ClipTypeBGElement	Psft_ClipTypeBG	
verb	VerbType	Create
[-] Psft_ClipType	Psft_ClipType	
ClipID	ClipIDType	CID12345
Clip	ClipType	clasp
Size	SizeType	small
Color	ColorType	black
[-] RetailUnits	Psft_RetailItem [ ]	
[-] RetailUnits[0]	Psft_RetailItem	
ItemCode	ItemCodeType	98765
Parkane	ParkaneType	hnx

- \_\_\_ c. Compare the data from the Visual Test Connector with the data in the Test Component to verify that the mapping operations you defined worked properly
- \_\_\_ 8. Stop and close the test component
- \_\_\_ a. Click the **Stop** (■) icon at the top right of the Events pane
- \_\_\_ b. Close the **PeopleSoftAdapter\_Test** component editor
- \_\_\_ c. Click **No** in the Save Resource dialog
- \_\_\_ 9. Close the Visual Test Connector
- \_\_\_ a. Select **File → Exit** in the VTC interface
- \_\_\_ b. Select **Yes** in the Shutdown dialog
- \_\_\_ c. Press **Enter** key In the Visual Test Connector command window to close it
- \_\_\_ 10. Remove the PeopleSoftAdapter module from the WebSphere Process Server test environment
- \_\_\_ a. Right click on WebSphere Process Server V6.0 from the Server view and select **Add and Remove Projects...** from the context menu
- \_\_\_ b. Select the **PeopleSoftAdapterApp** project from the Configured projects list
- \_\_\_ c. Click **< Remove**
- \_\_\_ d. Click **Finish**

## Part 5: Test the Siebel maps

As with PeopleSoft, there are two ways that the maps created for Siebel can be tested. In both cases the Interface Map is driven with the Test Component. The first approach would be to also use the Test Component to monitor the output of the Interface Map and the other would be to let it flow all the way to the Visual Test Connector. The later is the approach you will take in this section.

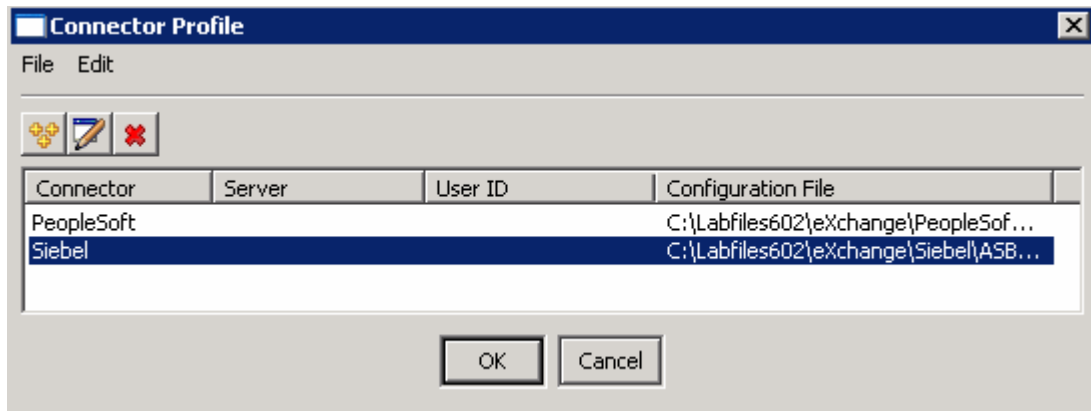
- \_\_\_ 1. Add the SiebelAdapter module to the WebSphere Process Server test environment
  - \_\_\_ a. Right click on WebSphere Process Server V6.0 from the Server view and select **Add and Remove Projects...** from the context menu
  - \_\_\_ b. Select only the **SiebelAdapterApp** project from the Available projects panel
  - \_\_\_ c. Click **Add >** to add it to the Configured projects panel
  - \_\_\_ d. Click **Finish**
- \_\_\_ 2. Connect the Visual Test Connector to the WebSphere Process Server test environment. Using the connector configuration at **<LAB\_FILES>\eXchange\Siebel\ASBOs\SiebelConnectorWPS.cfg** and follow the instructions below to use the Visual Test Connector:
- \_\_\_ 3. Ensure the WebSphere Process Server is active (the server is the JMS Broker). The VTC requires an active JMS Broker to start its connection.
  - \_\_\_ a. Ensure the WebSphere Process Server is active (the server is the JMS Broker). The VTC requires an active JMS Broker to start its connection
  - \_\_\_ b. Start the Visual Test Connector
  - \_\_\_ c. From the **start** menu, select **Programs → IBM WebSphere Business Integration Adapters → IBM WebSphere Business Integration Toolset → Visual Test Connector**. This opens a command window in the background where the process runs and the Visual Test Connector Interface in the foreground. Do not close the command window as this will cause the VTC to exit
  - \_\_\_ d. From the main menu of the VTC interface, select **Create/Select a Profile**. The Connector Profile panel opens
  - \_\_\_ e. Create a new Seibel profile by selecting **File → New Profile...** from the main menu of the Connector Profile panel

---

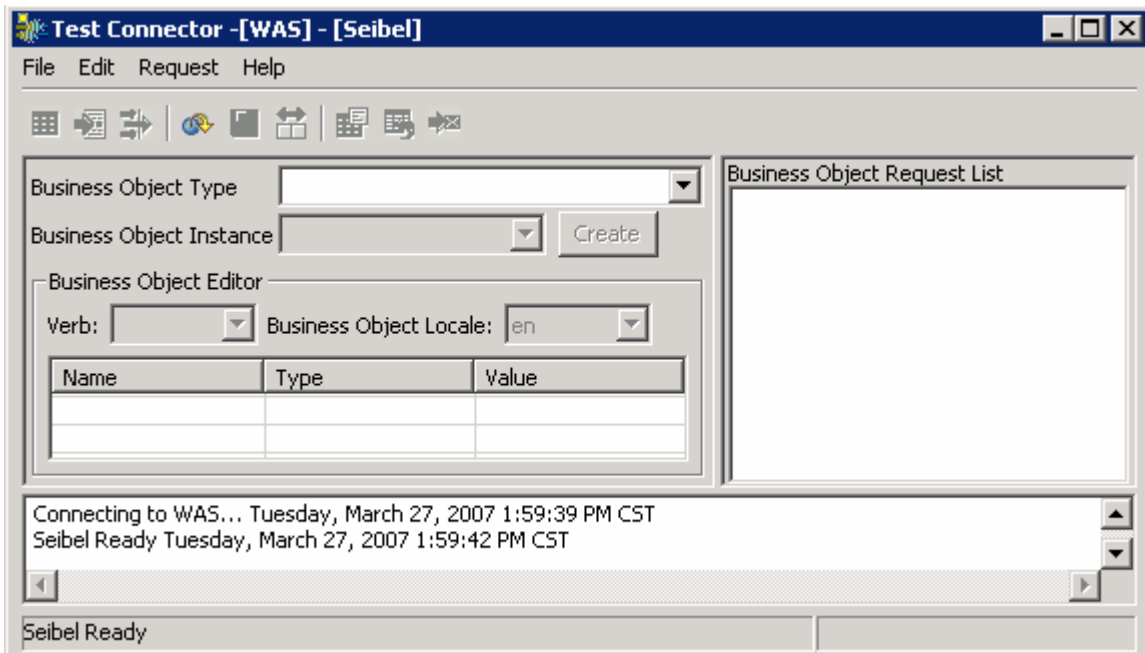
**Note:** You can skip these steps if a PeopleSoft profile has already been created and is listed.


---

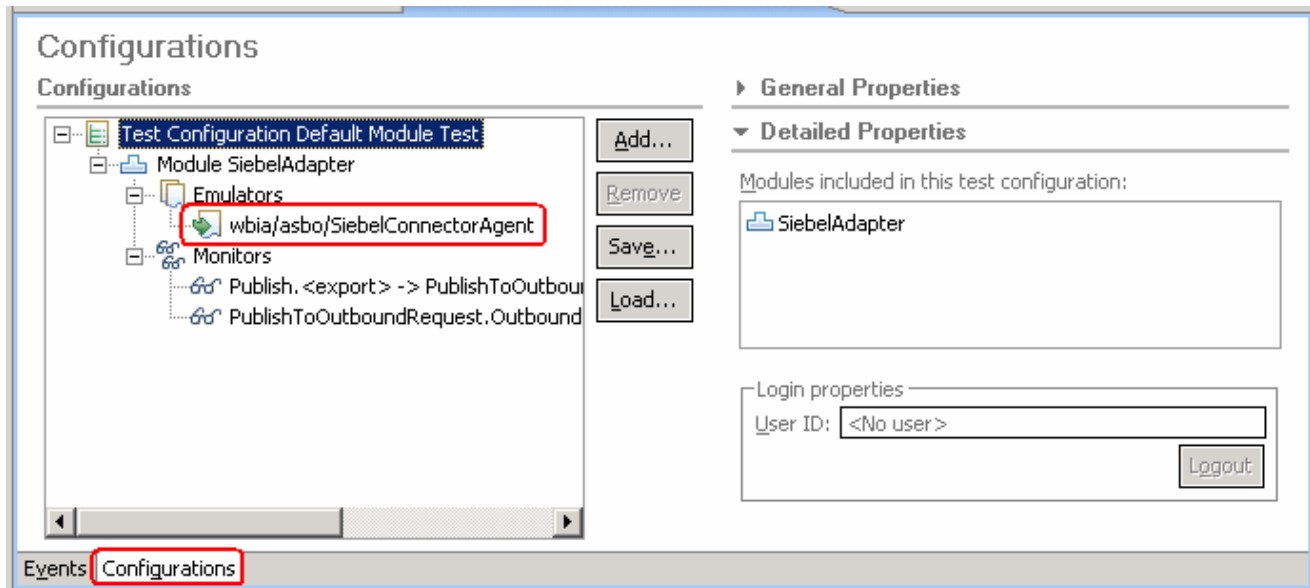
- \_\_\_ f. Enter the following information in the New Profile panel:
  - 1) Click the **Browse** button for "Select the connector configuration file:" filed and select **<LAB\_FILES>\eXchange\Siebel\ASBOs\SiebelConnectorWPS.cfg** file
  - 2) Connector Name : **Siebel**
  - 3) Broker Type : **WAS**
- \_\_\_ g. Click **OK**
- \_\_\_ h. Select the Seibel profile you had created as shown below:



- \_\_\_ i. Click **OK**
- \_\_\_ j. Now in the Test Connector interface, select **File → Connect** from the main menu
- \_\_\_ k. Ensure that a successful connection is made. You can confirm this by watching the trace messages displayed at the bottom frame of the Test Connector interface as shown below:



- \_\_\_ 4. In the WebSphere Integration Developer, expand **SiebelAdapter** in the Business Integration view and double click on **Assembly Diagram** (  **Assembly Diagram** ) to open it in an Assembly Diagram editor
- \_\_\_ 5. Setup the Test Component to drive the test
  - \_\_\_ a. In the Assembly Editor, right click on the **PublishToOutboundRequest** Interface Map and select **Test Component** from the context menu
  - \_\_\_ b. Click to select the **Configurations** tab at the bottom of the **PublishToOutboundRequest\_Test** panel. You should see something like in the following image:



\_\_\_ c. Notice that there is an Emulator defined for the SiebelConnectorAgent. This means the Test Component is set up to emulate the Import going to the Visual Test Connector rather than letting it process. Since you want to let this process, you need to remove this emulator

- 1) Select the emulator **wbia/asbo/SiebelConnectorAgent**
- 2) Click the **Remove** button

\_\_\_ d. Click on the **Events** tab at the bottom of the **PublishToOutboundRequest\_Test** panel

\_\_\_ e. Enter the following values for Initial Request parameters:

- 1) verb : Select **Create** from the drop down list
- 2) clipID: **CID12345**
- 3) GLN: **1234567890123**
- 4) clip: **clasp**
- 5) size: **small**
- 6) color: **black**
- 7) brand: **Clips and Tacks**

\_\_\_ f. Create a ClipItem instance

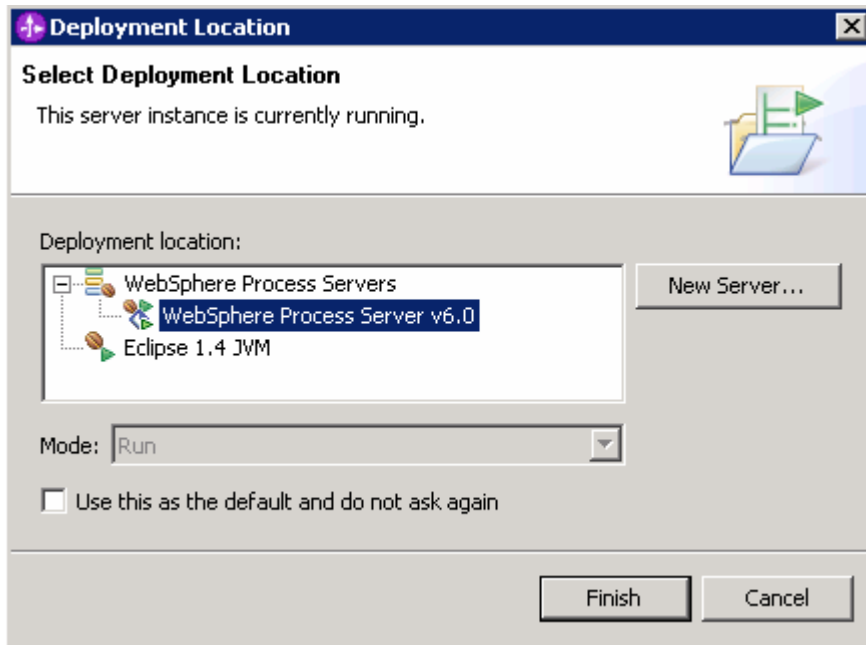
- 1) Select and right click on **retailItems**
- 2) Select **Add Element** from the context menu
- 3) Enter **1** for the "Number of New Elements to Add" in the New Element dialog and click **OK**

\_\_\_ g. Now enter the following values under **retailItems**

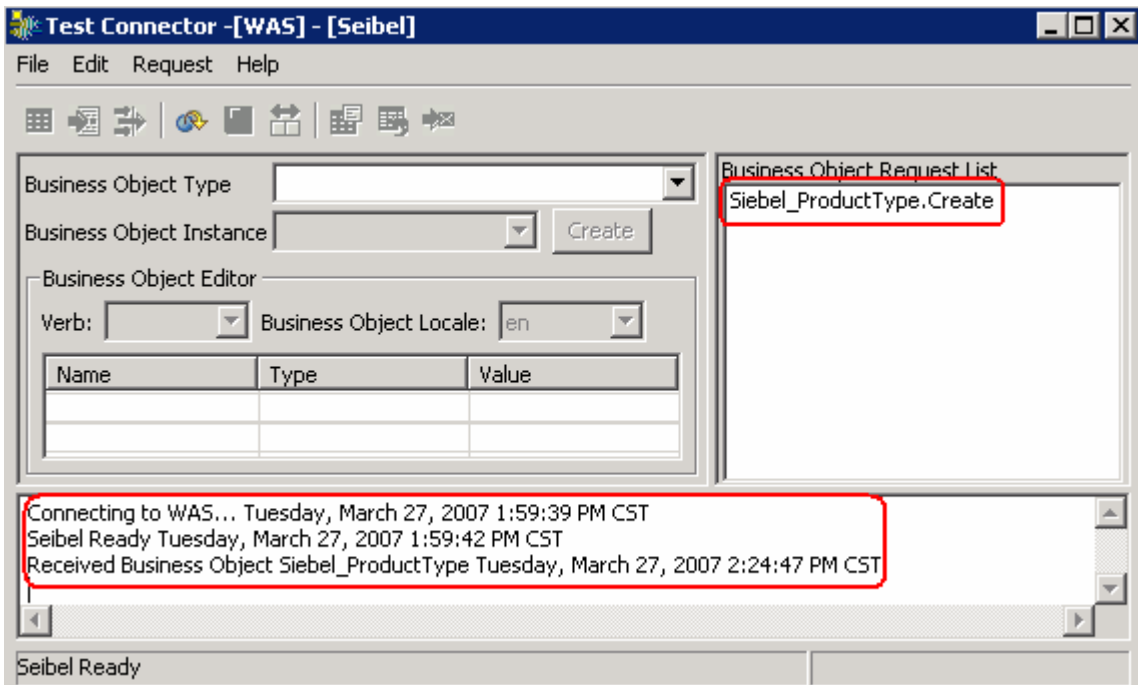
- 1) itemID: **98765**

- 2) GTIN: **12345678912345**
- 3) package: **box**
- 4) quantity: **40**
- 5) fullDescription: **box 40**
- 6) price: **2.59**
- 7) startDate: **2007-27-01**
- 8) endDate: **2010-06-30**
- 9) contactFirstName: **Sam**
- 10) contactLastName: **Jones**

- \_\_\_ 6. Run the test
  - \_\_\_ a. Click **Continue**
  - \_\_\_ b. Select the **WebSphere Process Server V6.0** server in the **Choose a deployment location** dialog

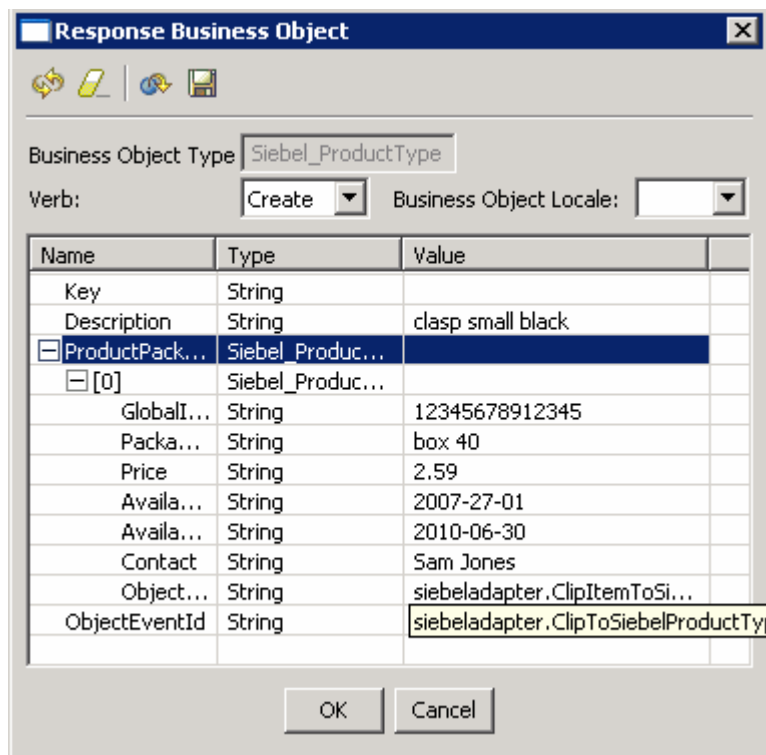


- \_\_\_ c. Click **Finish**
- \_\_\_ d. Review the Visual Test Connector window to see if a message has been received



\_\_\_ e. Double click on **Siebel\_ProductType.create** in the Business Object Request List to view the business object received by the Visual Test Connector

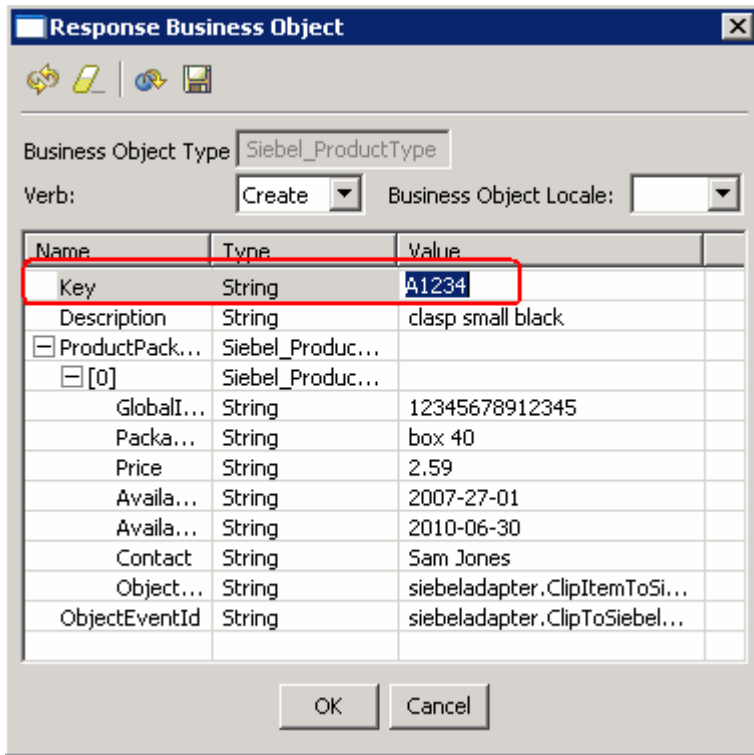
\_\_\_ f. The business object should look like this:



\_\_\_ g. Compare the values reported above with those entered in the Test Component to see if the mapping definitions you defined appear to have worked correctly. Also, these should be the same as what was passed on the wire

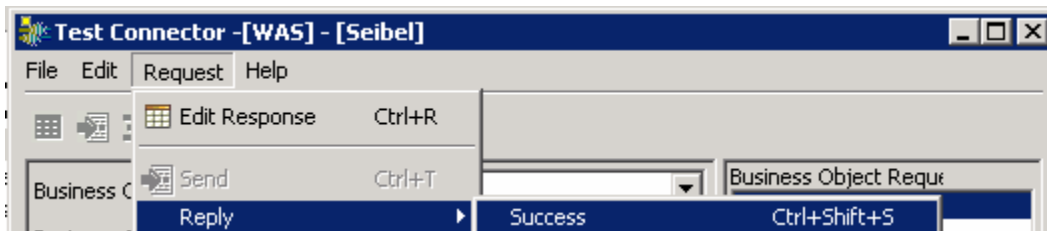
\_\_\_ 7. Now you can send back a response to the server

\_\_\_ a. In the Response Business Object window you will enter a value for the **Key** as **A1234**



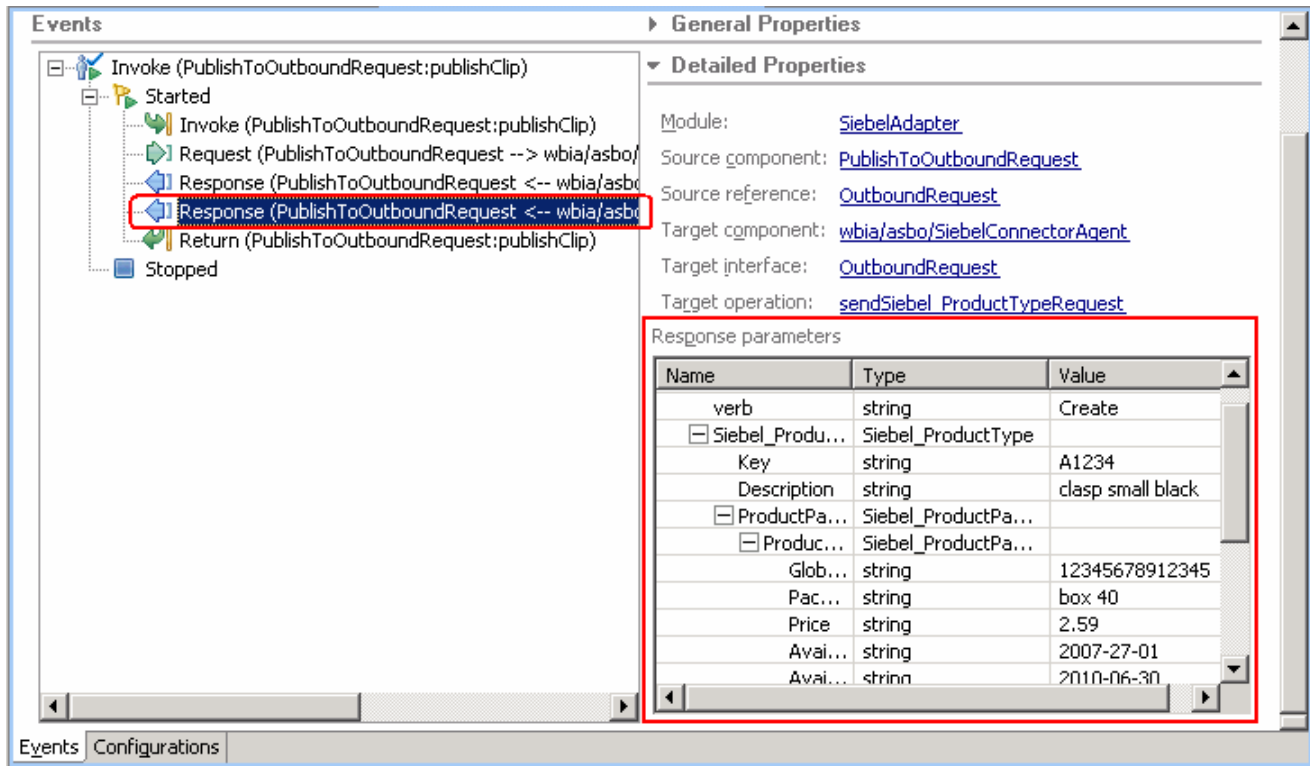
\_\_\_ b. Click **OK**

\_\_\_ c. Send the response back to the server by selecting **Request** → **Reply** → **Success** Visual Test Connector window



\_\_\_ d. In the WebSphere Integration Developer, you will see a Response to the Invocation in the Test Component. Double click on the Response so that the Response parameters show on the right panel. Examine the returned ebel\_ProductType business object to see the Key has been set to A1234





- \_\_\_ 8. Stop and close the test component
  - \_\_\_ a. Click the **Stop** (■) icon at the top right of the Events pane
  - \_\_\_ b. Close the Test component editor
  - \_\_\_ c. Click **No** in the Save Resource dialog
- \_\_\_ 9. Close the Visual Test Connector
  - \_\_\_ a. Select **File → Exit** In the VTC window
  - \_\_\_ b. Select **Yes** in the Shutdown dialog
  - \_\_\_ c. Click the **Enter** key In the Visual Test Connector command window to close it
- \_\_\_ 10. Clean WebSphere Process Server test environment
  - \_\_\_ a. Right-click on WebSphere Process Server V6 from the Servers view and select **Add and remove projects...** from the context menu
  - \_\_\_ b. Click **<< Remove All**
  - \_\_\_ c. Click **Finish**
- \_\_\_ 11. Stop the Server
  - \_\_\_ a. Right click on WebSphere Process Server V6.0 server from the Servers view and select **Stop** from the context menu

## **What you did in this exercise**

You defined Interface Maps and Data Maps which were used to map between ASBOs and GBOs. You tested these maps using the Component Test capabilities in WebSphere Integration Developer.

## Solution instructions

Follow the directions in the task [Initialize the Workspace for a Lab Exercise](#), using the following values:

**<WORKSPACE>**

C:\Labfiles602\Exchange\WBIAdapters\workspaceMaps

**<PROJECT\_INTERCHANGE>**

C:\Labfiles602\Exchange\WBIAdapters\Solution\Mapping\_PI.zip

**<MODULE>**

n/a

**<DEPENDENT\_LIBRARIES>**

n/a

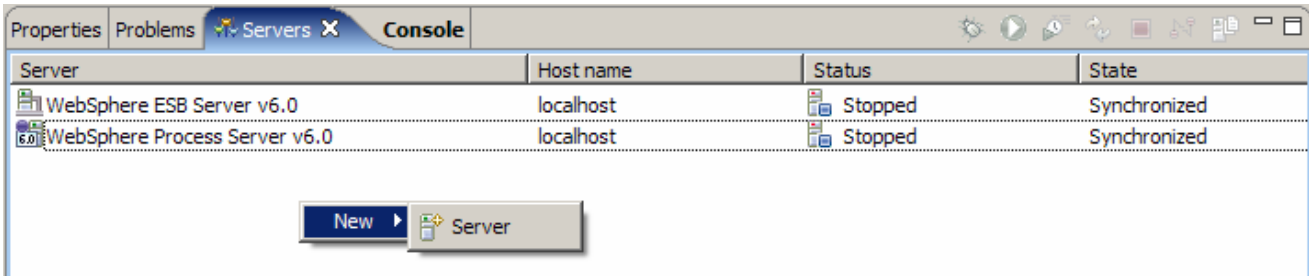
Continue with Part 4: Test the PeopleSoft Maps.

# Task: Adding remote server to WebSphere Integration Developer test environment

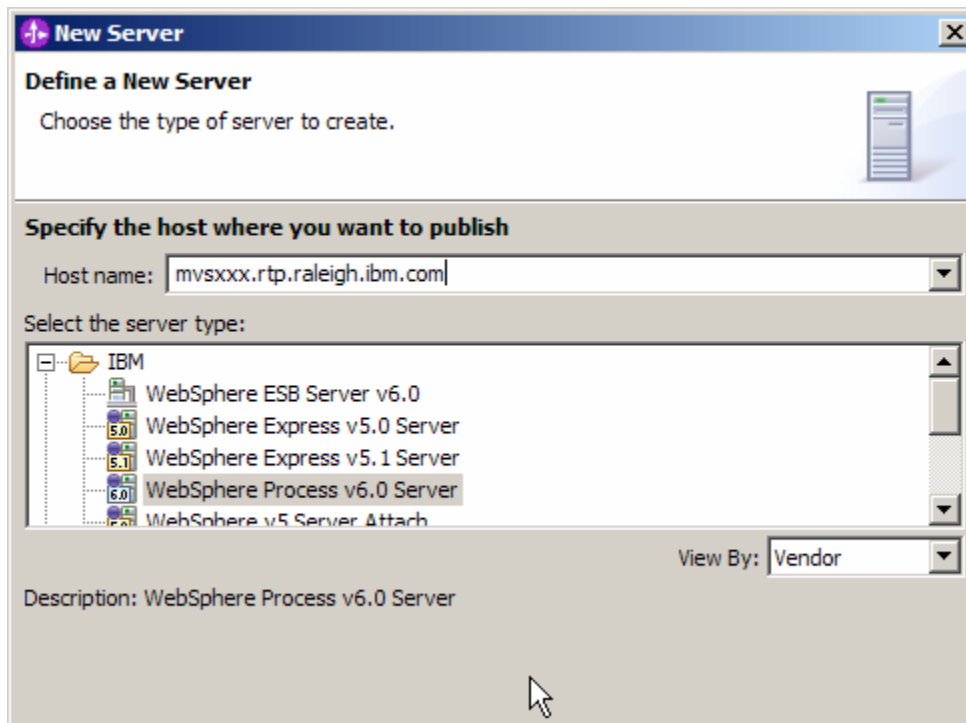
This task describes how to add a remote server to the WebSphere Integration Developer Test environment. The sample you will use is a z/OS machine.

Create a new remote server.

- \_\_\_ 1. Right click on the background of the Servers view to access the pop-up menu.
- \_\_\_ 2. Select **New → Server**.



- \_\_\_ 3. Specify host name to the remote server, **<HOSTNAME>**.
- \_\_\_ 4. Ensure that 'WebSphere Process V6.0 Server' is highlighted in the server type list.



- \_\_\_ 5. Click **Next**.

- \_\_\_ 6. On the WebSphere Server Settings page, select the radio button for **RMI** and change the ORB bootstrap port to the correct setting (<BOOTSTRAP\_PORT>).

**New Server**  
**WebSphere Server Settings**  
Input settings for the new WebSphere server

WebSphere profile name: [ ]

Server connection type and admin port

RMI (Better performance)  
ORB bootstrap port: [ 9131 ]

SOAP (More firewall compatible)  
SOAP connector port: [ 8880 ]

Run server with resources within the workspace

Security is enabled on this server

Current active authentication settings:

User ID: [ ]

Password: [ ]

Server name: [ server1 ]

Server type

BASE, Express or unmanaged Network Deployment server

Network Deployment server

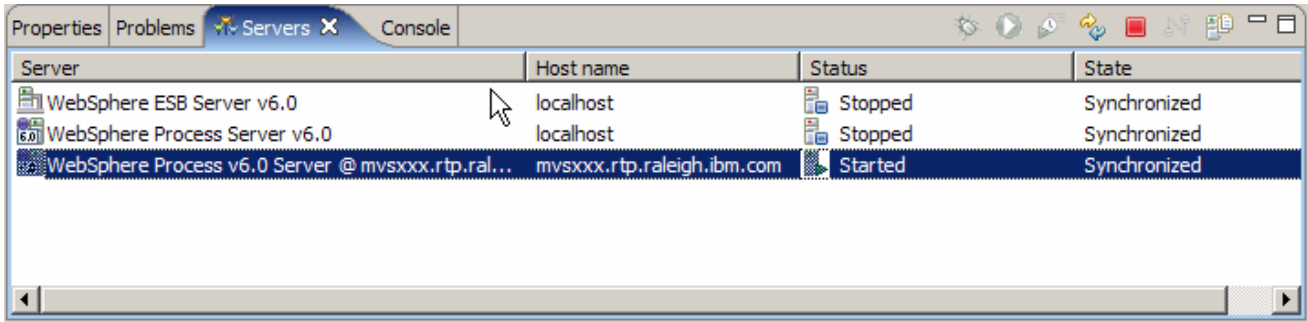
Network Deployment server name: [ ]

The server name is in the form of:  
<cell name>/<node name>/<server name>  
For example, localhost/localhost/server1.

[ Detect ] Click this button to detect the server type.

[ < Back ] [ Next > ] [ Finish ] [ Cancel ]

- \_\_\_ 7. Click **Finish**.
- \_\_\_ 8. The new server should be seen in the Server view.



- \_\_\_ 9. Start the remote server if it is not already started. WebSphere Integration Developer does not support starting remote servers from the Server View.
- \_\_\_ 10. From a command prompt, telnet to the remote system if needed:
- ```
'telnet <HOSTNAME> <TELNET_PORT>'
      userid : <USERID>
      pw : <PASSWORD>
```
- \_\_\_ 11. Navigate to the bin directory for the profile being used:
- ```
cd <WAS_HOME>/profiles/<PROFILE_NAME>/bin
```
- \_\_\_ 12. Run the command file to start the server: `./startServer.sh <SERVER_NAME>`
- \_\_\_ 13. Wait for status message indicating server has started:

```
ADMU3200I: Server launched. Waiting for initialization status
ADMU3000I: Server c11sr01 open for e-business; process id is 0000012000000002
```