

Maintaining relationships between ASBOs

What this exercise is about	2
Lab requirements	2
What you should be able to do	2
Introduction	3
Exercise instructions	4
Part 1: Initialize the workspace for this lab exercise	6
Part 2: Authoring relationships	8
Part 3: Completing the relationship picture	17
Part 4: Initialize the workspace for the testing exercise	25
What you did in this exercise	36
Solution instructions	37
Task: Adding remote server to WebSphere Integration Developer test environment	38

What this exercise is about

This exercise shows you how to develop and test relationships. These are used to cross-reference the PeopleSoft ASBOs with the GBOs and from the GBOs to the Siebel ASBOs.

Lab requirements

List of system and software 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

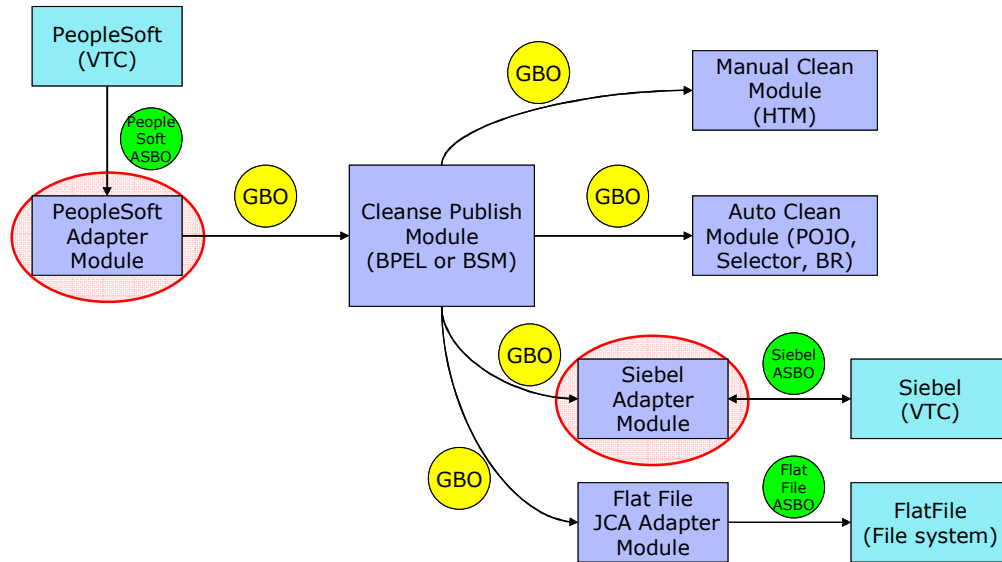
What you should be able to do

At the end of this lab you should be able to

- Design, test and utilize relationships
 - Create a relationship definition and relationship roles
 - Test the relationships using Test Component

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 – Mapping** lab.



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\workspaceRel	/tmp/Labfiles602/eXchange//WBIAdapters\workspaceRel
<TEMP>	C:\temp	/tmp

Windows users: 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 [®] 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[®], AIX[®] 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

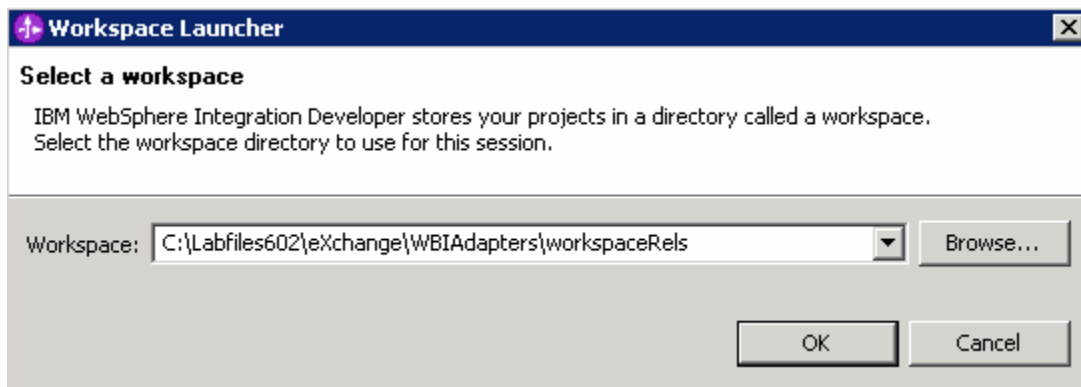
This lab exercise is dependent upon the completion of the **WBIAdapters – Mapping between ASBOs and GBPS** lab exercise. There are two approaches you can take to performing this lab.

1. **Import pre-built projects** – Using this approach, you will import a project which will establish your workspace with the required constructs from the previous lab exercise.
2. **Use your existing workspace** – Using this approach, you will continue to use the workspace you were using in the previous lab exercise.

It is recommended that you use approach 1 where you import pre-built projects. This is because it prevents possible problems that might have been introduced when you completed the previous lab. In addition, the testing of the previous lab leaves some testing specific constructs that are not needed in this lab.

Perform **only one** of the following steps depending upon your choice of workspace.

- ___ 1. Start WebSphere Integration Developer V6.0.2 with a workspace location of **<WORKSPACE>**, that is. **<LAB_FILES>\eXchange\WBIAdapters\workspaceRels**
 - ___ 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**



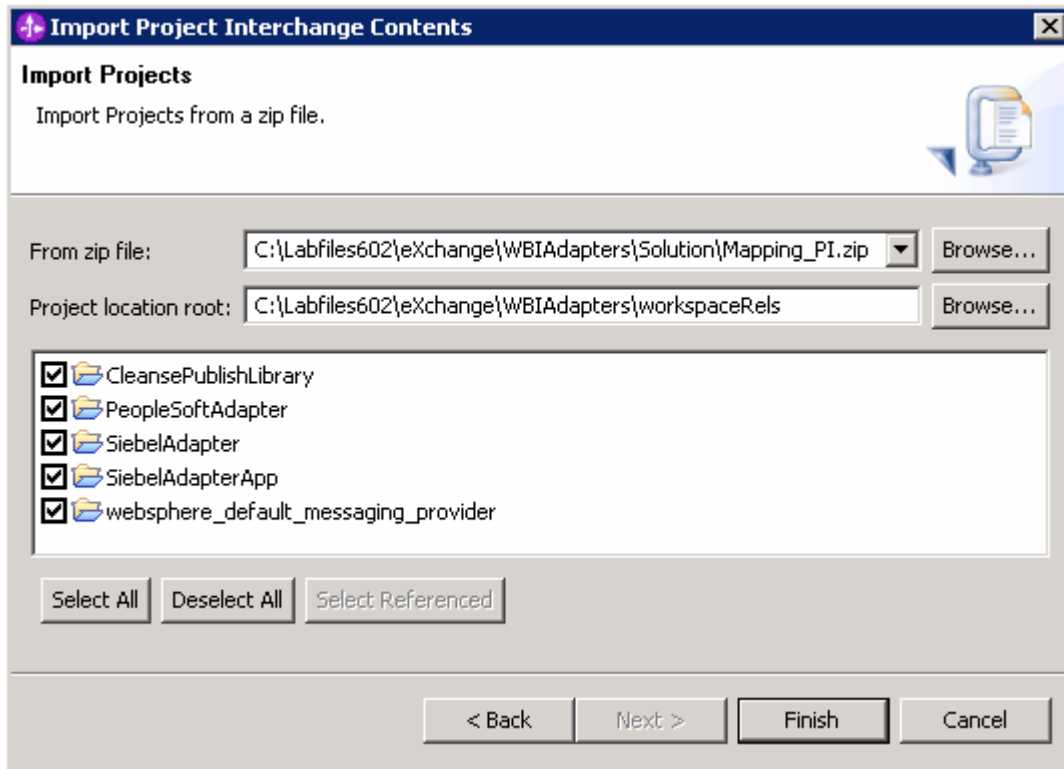
- ___ c. When WebSphere Integration Developer V6.0.2 opens, click the curved arrow at top right to **go to Business Integration perspective**



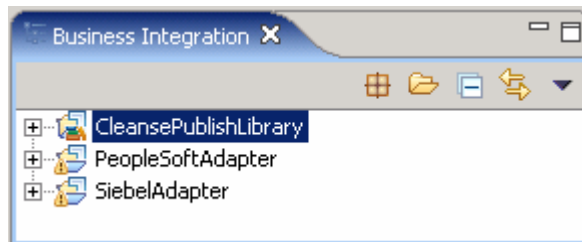
- ___ d. Ensure you are in the **Business Integration** perspective
- ___ 2. Import Project Interchange file, ImportExport_Pi.zip located at **<LAB_FILES>\eXchange\WBIAdapters\Solution\Mapping_Pi.zip**

Note: As this lab is dependent upon **WBIAdapters – Mappings** 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\Mapping_PI.zip** and hit **Open**



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

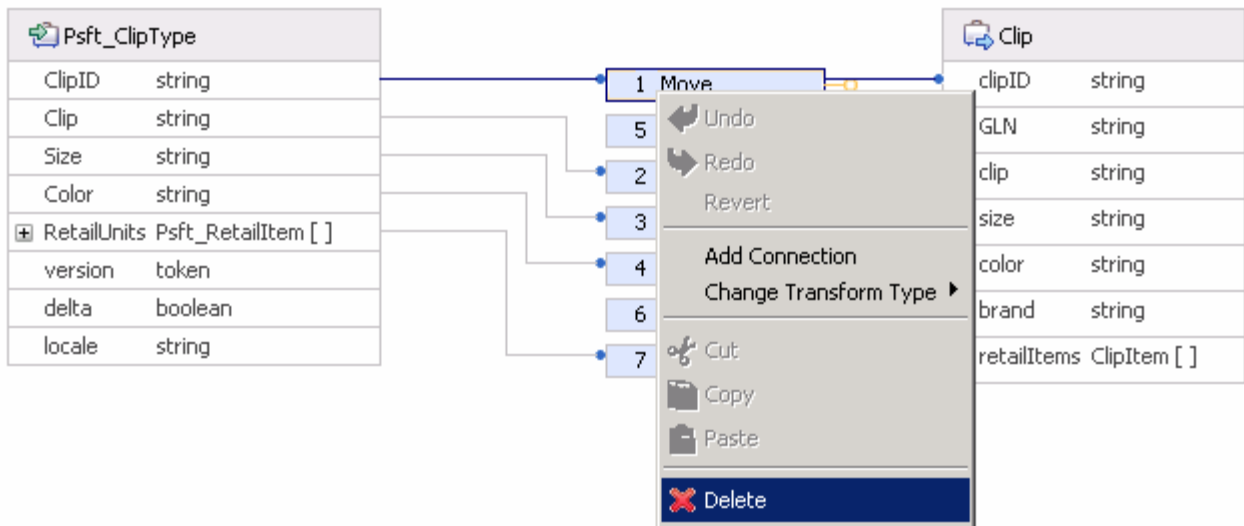


- ___ 3. Verify you have WebSphere Process Server V6.0 listed in your Servers view

Part 2: Authoring relationships

In this section you will modify the maps created in the previous lab to contain an identity relationship that will associate a clip type from PeopleSoft with a product type in Siebel. Logically, the relationship will be between the ClipID field in Psft_ClipType and the Key field in Siebel_ProductType. To do this, the relationships are actually tied together using the clipID from the Clip generic business object and defining it as managed. This then becomes the link through which the Relationship Service ties together ClipID in Psft_ClipType and Key in Siebel_ProductType.

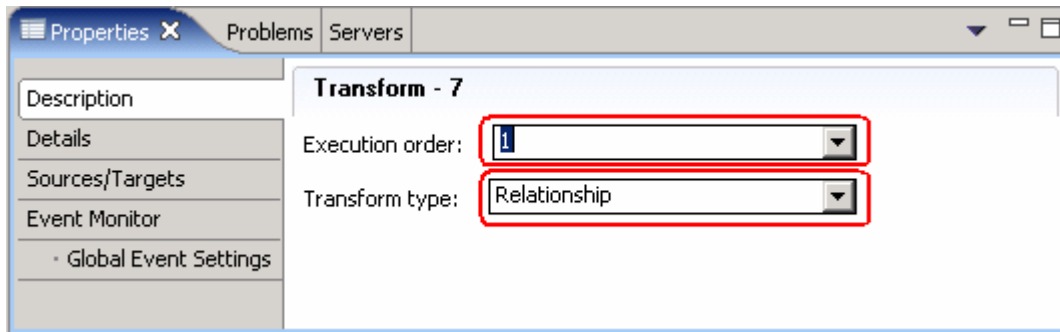
- ___ 1. Create a relationship between **ClipID** in **Psft_ClipType** and **clipID** in **Clip**. To do this, you must change the already existing Move transformation for these fields to a Relationship transformation
 - ___ a. In the Business Integration view, expand **PeopleSoftAdapter** → **Mapping** → **Data Maps**
 - ___ b. Double-click on **PsftClipTypeToClip** to open it in the Mapping Editor
 - ___ c. Right click on the Move between **ClipID** and **clipID** and select **Delete**



- ___ d. Draw a line from **Psft_ClipType** to **Clip**
- ___ e. Click on the left side of the **Submap** transformation rule to view the properties in the Properties view



- ___ f. Click to select the **Description** tab in the **Properties** view as shown below:
 - 1) Using the **Transform type** dropdown, change the transformation rule setting from **Submap** to **Relationship**
 - 2) Use the **Execution order** dropdown and change its setting to **1** for the Relationship rule

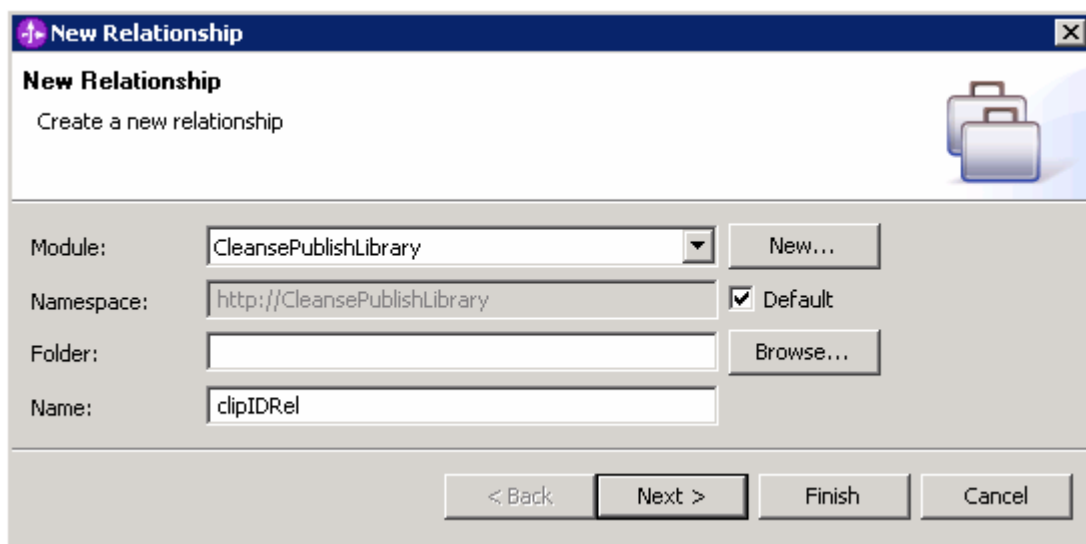


___ 2. Click on the **Details** tab

___ a. Click the **New...** button by the Relationship Definition field to open the New Relationship dialog

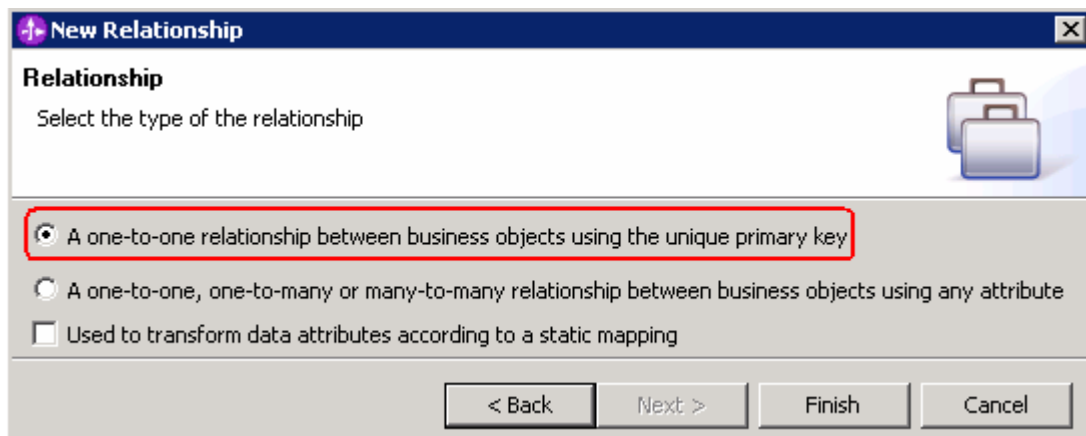
1) In the New Relationship dialog, select **CleansePublishLibrary** from the **Module** dropdown

2) Type in a **Name** for the new relationship definition: **clipIDRel**



___ b. Click **Next**

___ c. In the following screen, ensure that the radio button next to **one-to-one relationship** is selected




___ d. Click **Finish**. The relationship definition editor is opened

Note: You might notice that the relationship is now flagged with an error. This is because no Relationship Roles have been defined yet. This error will be resolved shortly.


___ 3. Define the roles in the relationship

___ a. If you closed the **clipIDRel** component, navigate to **CleansePublishLibrary → Mapping → Relationships** in the Business Integration view and double click on **clipIDRel** to open it

___ b. Add a new relationship role by clicking on the **Add Role** () icon



___ c. In the Data Type Selection dialog, select **Clip** and click **OK**

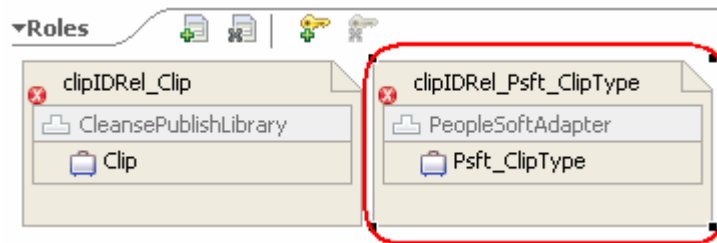
___ d. Add a new relationship role by clicking on the **Add Role** () icon

___ e. Select **Psft_ClipType** in the Data Type Selection dialog

___ f. Click **OK**

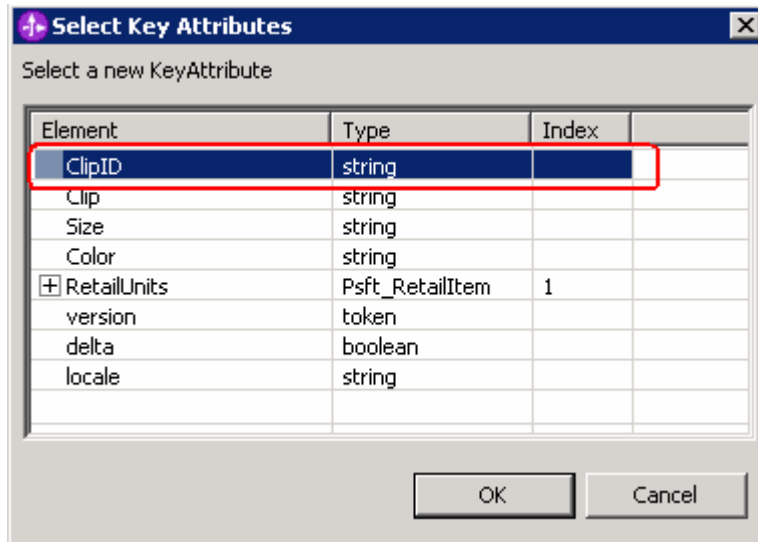
___ g. Press **Ctrl+S** to save the relationship definition

___ h. Select the **clipIDRel_Psft_ClipType** Role



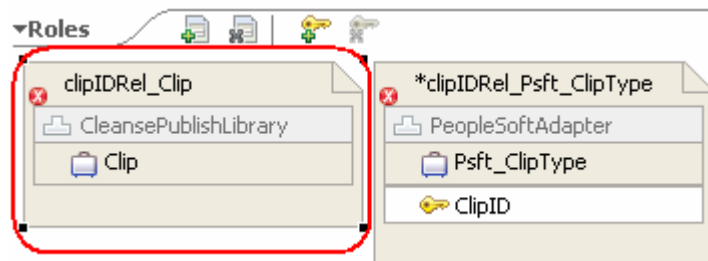
___ i. Click on the **Add Key Attribute** () icon

___ j. Select **ClipID** in the Select Key Attributes dialog



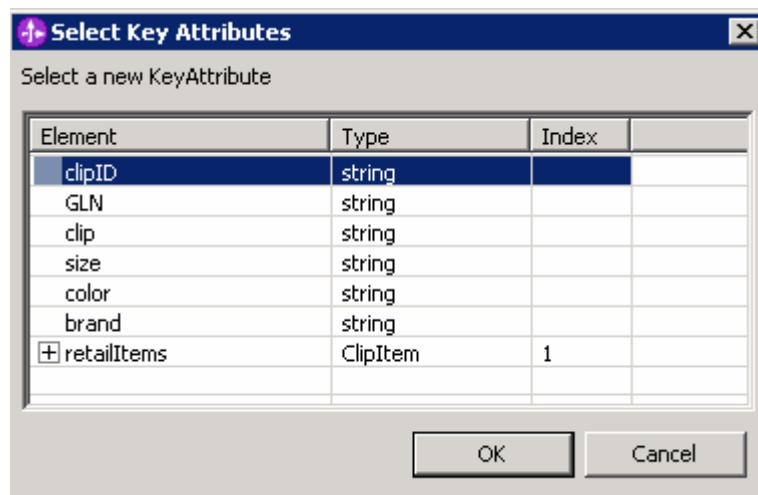
__ k. Click **OK**

__ l. Click on the **clipIDRel_Clip** box to select it



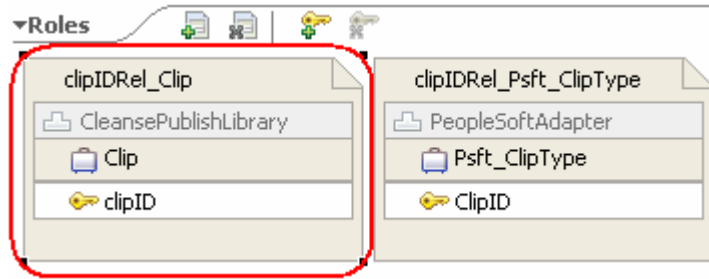
__ m. Click on the **Add Key Attribute** (🔑) icon

__ n. Select **clipID** in the Select Key Attributes dialog

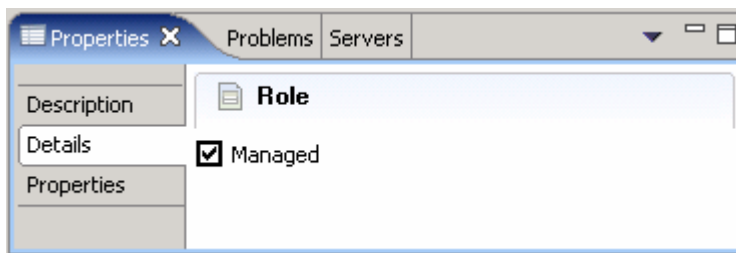


__ o. Click **OK**

__ p. Press **Ctrl+S** to save the relationship definition



- __ q. Ensure that the **clipIDRel_Clip** box is still selected to access its properties in the Properties view
- __ r. Click to select the **Details** tab in the **Properties** view
- __ s. Select the check box next to **Managed**

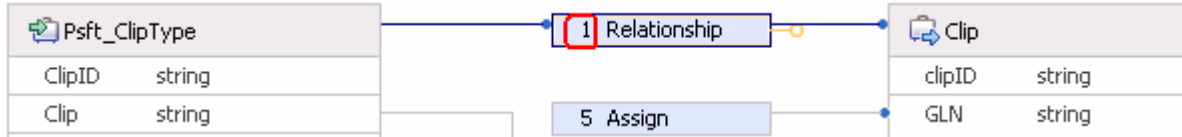


- __ t. Press **Ctrl+S** to save the relationship definition and you can now close it
- __ u. The final relation definition looks like as shown below:



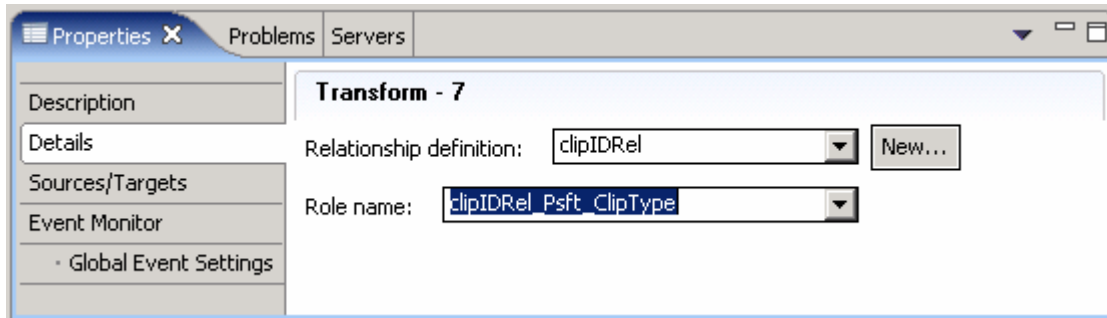
Note: There must not be any errors reflected in the Problems view at this time.

- ___ 4. Add a new relationship role to the Relationship transformation in a map
 - __ a. Switch to the map by clicking on the **PsftClipTypeToClip** tab in the top panel
 - __ b. Select the left side of the **Relationship** transform



__ c. Click to select the **Details** tab in the **Properties** view

__ d. Use the dropdown list box to select **clipIDRel_Psft_ClipType** for the **Role Name**



__ e. Press **Ctrl+S** to save the map and close it

___ 5. Create a relationship between **clipID** in **Clip** and **Key** in **Siebel_ProductType**

__ a. In the Business Integration view, expand **SiebelAdapter** → **Mapping** → **Data Maps**

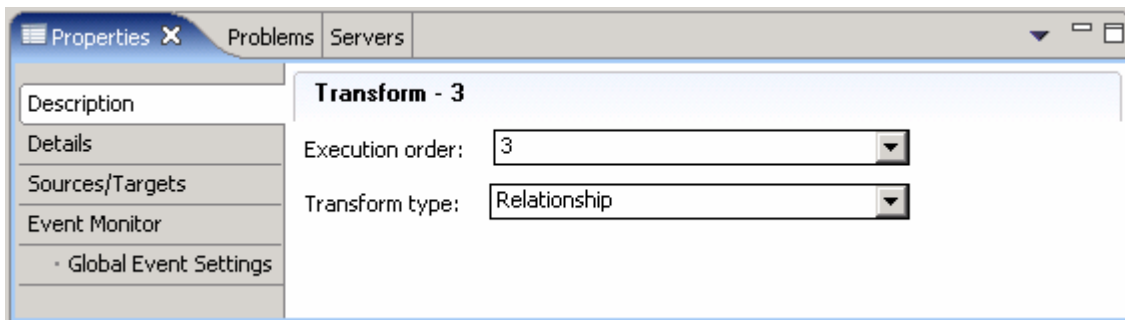
__ b. Double click on **ClipToSiebelProductType** to open it in the Mapping Editor

__ c. Draw a line from **Clip** to **Siebel_ProductType**

__ d. Click on the left side of the **Submap** transformation rule label

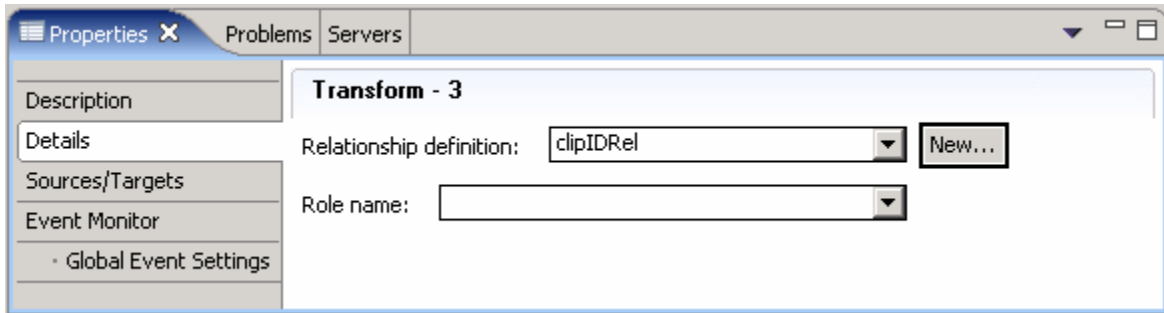
__ e. Click to select the **Description** tab in the Properties view

__ f. Change the rule from Submap to **Relationship** using the transformation dropdown list



__ g. Now select the **Details** tab in the **Properties** tab

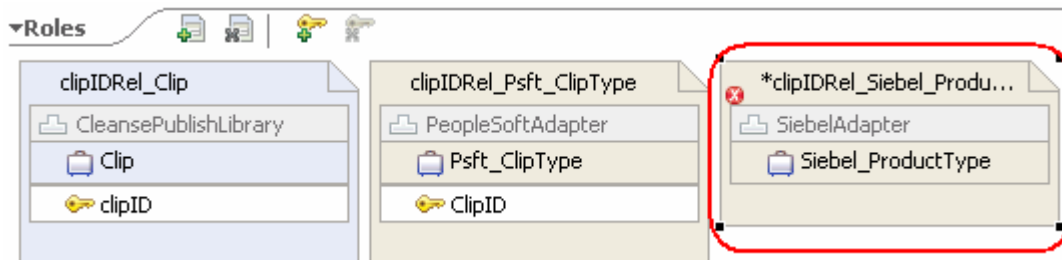
__ h. Select **clipIDRel** from the drop down list of the Relationship definition field



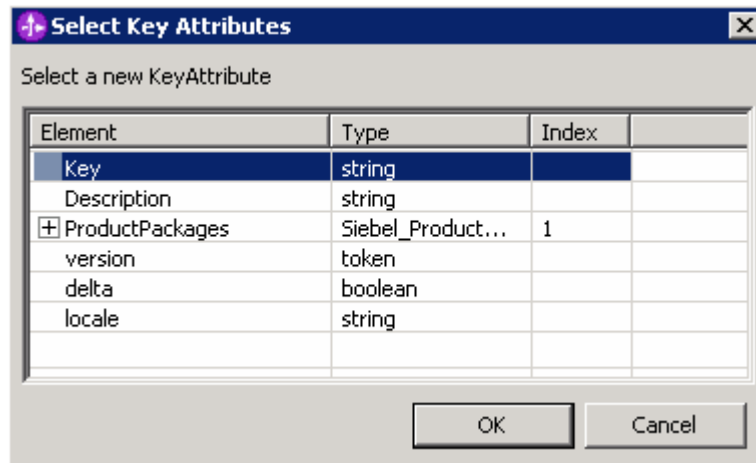
- ___ 6. Define the roles in the relationship
 - ___ a. In the Business Integration view, expand **CleansePublishLibrary** → **Mapping** → **Relationships**
 - ___ b. Double click on **clipIDRel** to open it in the Relationship editor



- ___ c. Add a new relationship role by clicking on the **Add Role** (📄+) icon
- ___ d. Select **Siebel_ProductType** in the Data Type Selection dialog
- ___ e. Click **OK**
- ___ f. Click on the new **ClipIDRel_Siebel_ProductType** role to select it



- ___ g. Click on the **Add Key Attribute** (🔑+) icon
- ___ h. Select **Key** in the Select Key Attributes dialog



___ i. Click **OK**

___ j. Press **Ctrl+S** to save the relationship definition and close the relationship definition editor

___ 7. Add the new relationship role to the map

___ a. Switch to the map by clicking on the **ClipToSiebelProductType** tab in the top panel

___ b. Select the left side of the **Relationship** transformation rule

___ c. Click to select the **Description** tab in the Properties view

___ d. Change the **Execution order** setting to **1** for the Relationship rule from the drop down

___ e. Now select the **Details** tab in the Properties view

___ f. Use the drop down box to select **clipIDRel Siebel_ProductType** for the Role Name

The screenshot displays the IBM Business Process Manager (BPM) interface. At the top, the window title is '*ClipToSiebelProductType'. Below this, the 'Business object map' section shows a single object 'ClipToSiebelProductType'. The 'Transformations' section contains three transformations: '1 Relationship', '2 Join', and '3 Submap'. The 'Clip' object has attributes: clipID (string), GLN (string), clip (string), size (string), color (string), brand (string), and retailItems (ClipItem []). The 'Siebel_ProductType' object has attributes: Key (string), Description (string), and ProductPackages (Siebel_ProductPackage []). The 'Properties' window at the bottom is open to the 'Transform - 3' tab. It shows the 'Relationship definition' as 'clipIDRel' and the 'Role name' dropdown menu with 'clipIDRel_Siebel_ProductType' selected. Other options in the dropdown are 'clipIDRel_Psft_ClipType' and 'clipIDRel_Siebel_ProductType'.

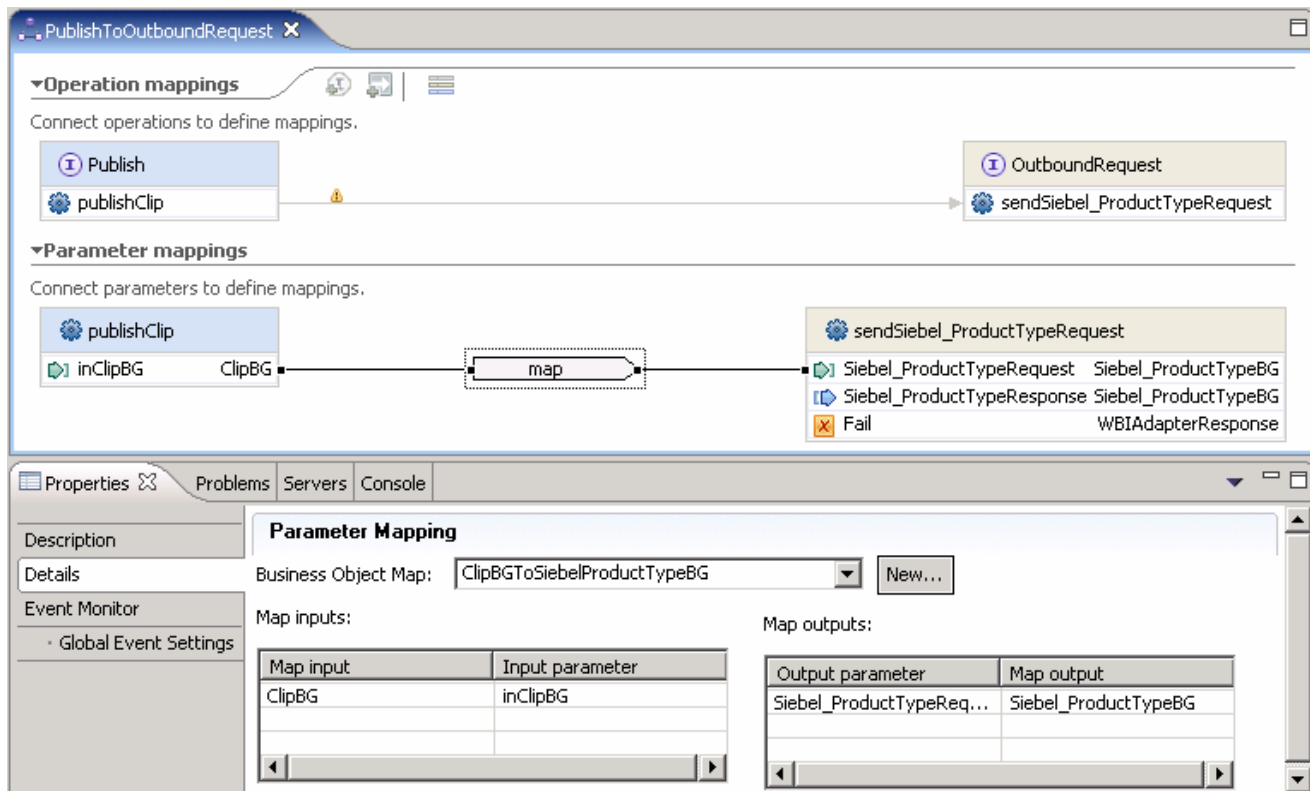
___ 8. Press **Ctrl+S** to save the modified map and close the map definition editor

Part 3: Completing the relationship picture

What you did in the previous section showed you how to go through the authoring process for a Relationship. However, there is more work that is required to complete the relationship story. In this section you examine what additional work is needed without having you go through the actual authoring steps needed to create the rest of the artifacts. Then in Part 4 you will import them and run the tests.

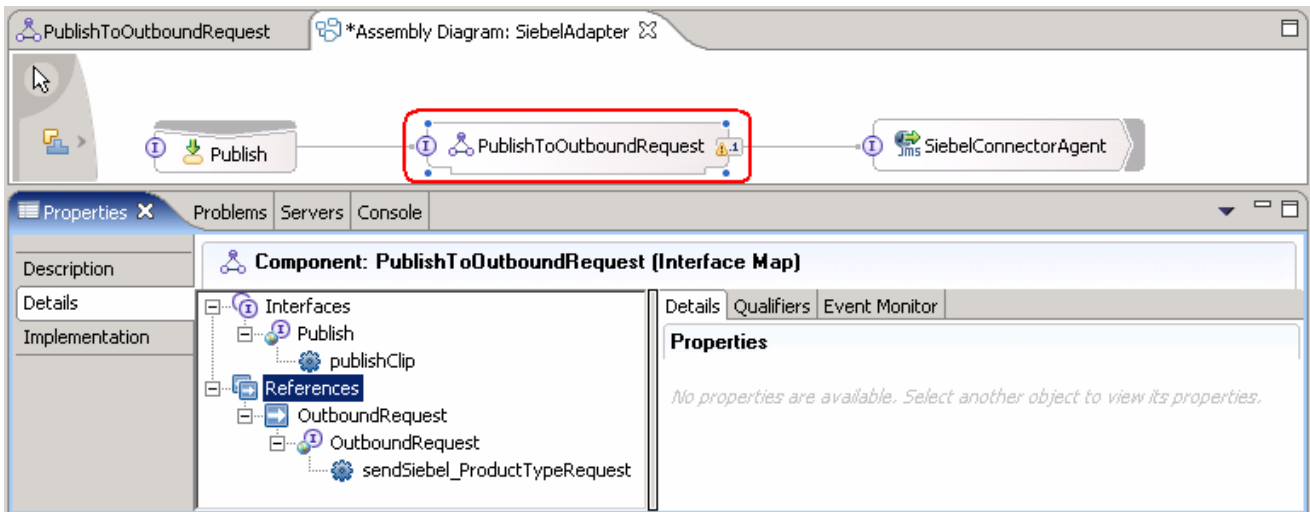
Since you do not import these until Part 4, you will not see them in WebSphere Integration Developer at this time. Simply read through Part 3 to understand what is required.

Start by looking at the **PublishToOutboundRequest** interface map that you created in the previous lab on mapping. (**SiebelAdapter** → **Mapping** → **Interface Maps**)



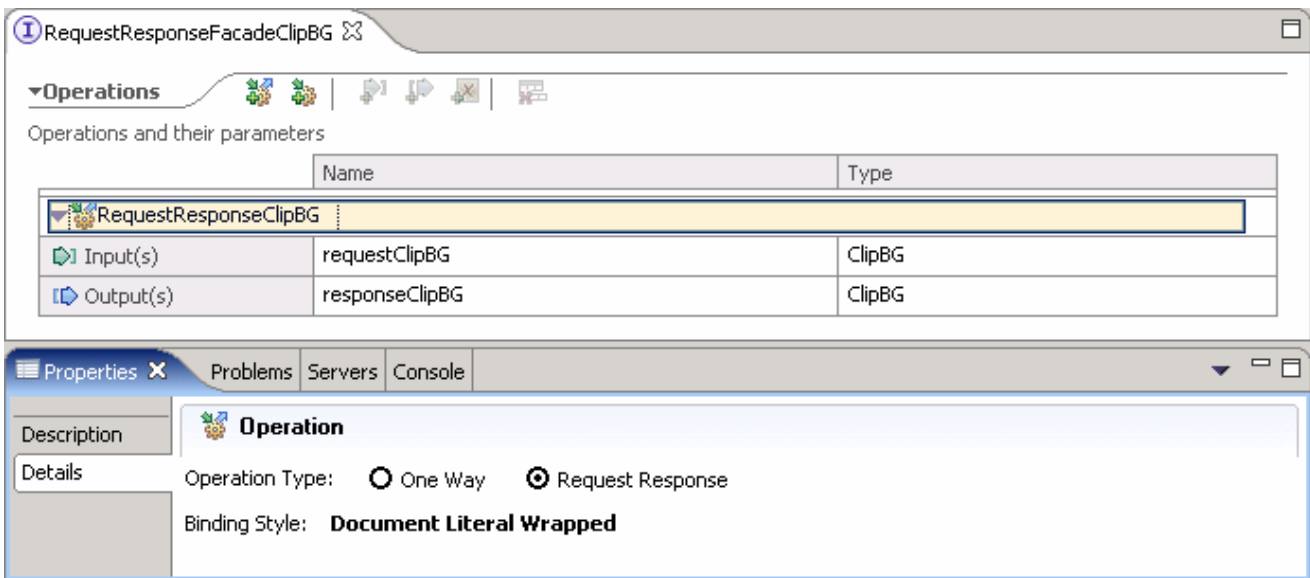
The inClipBG parameter is mapped to the **Siebel_ProductTypeRequest** parameter on the request. However, the **Siebel_ProductTypeResponse** parameter is not mapped to anything on the response. In a create scenario, there is no key associated with the **Siebel_ProductType** business object until the response comes back from Siebel. There must be a mapping operation to map the Siebel_ProductType being returned to a Clip. It is during this mapping operation that the relationship is updated with the new key value.

Probably the best way to start explaining this would be to start with the wiring diagram. This is what your wiring diagram currently looks like. (Open the SiebelAdapter Assembly Diagram, **SiebelAdapter** → **Assembly**)



First, you are going to need three data maps to map between the Siebel ASBOs and the GBOs. In order for those maps to be invoked, the **PublishToOutboundRequest** interface map must be able to return a ClipBG. This will require a new interface on the input side. The introduction of the new interface will force another interface map to be introduced to map between the Publish interface and itself. So the list of artifacts needed would be:

- Interface
 - **RequestResponseFacadeClipBG** – has ClipBG as input & output parameters.



- Interface Maps
 - **RequestResponseFacadeToOutboundRequest** – replaces the existing **PublishToOutboundRequest** interface mapping.

RequestResponseFacadeToOutboundRequest

Operation mappings
Connect operations to define mappings.

RequestResponseFacadeClipBG → OutboundRequest
RequestResponseClipBG → sendSiebel_ProductTypeRequest

Parameter mappings
Connect parameters to define mappings.

RequestResponseClipBG → sendSiebel_ProductTypeRequest

requestClipBG (ClipBG) → map → Siebel_ProductTypeRequest (Siebel_ProductTypeBG)
responseClipBG (ClipBG) → map → Siebel_ProductTypeResponse (Siebel_ProductTypeBG)
Fail → WBIAdapterResponse

Parameter Mapping

Business Object Map: ClipBGToSiebelProductTypeBG

Map input	Input parameter	Output parameter	Map output
ClipBG	requestClipBG	Siebel_ProductTypeReq...	Siebel_ProductTypeBG

RequestResponseFacadeToOutboundRequest

Operation mappings
Connect operations to define mappings.

RequestResponseFacadeClipBG → OutboundRequest
RequestResponseClipBG → sendSiebel_ProductTypeRequest

Parameter mappings
Connect parameters to define mappings.

RequestResponseClipBG → sendSiebel_ProductTypeRequest

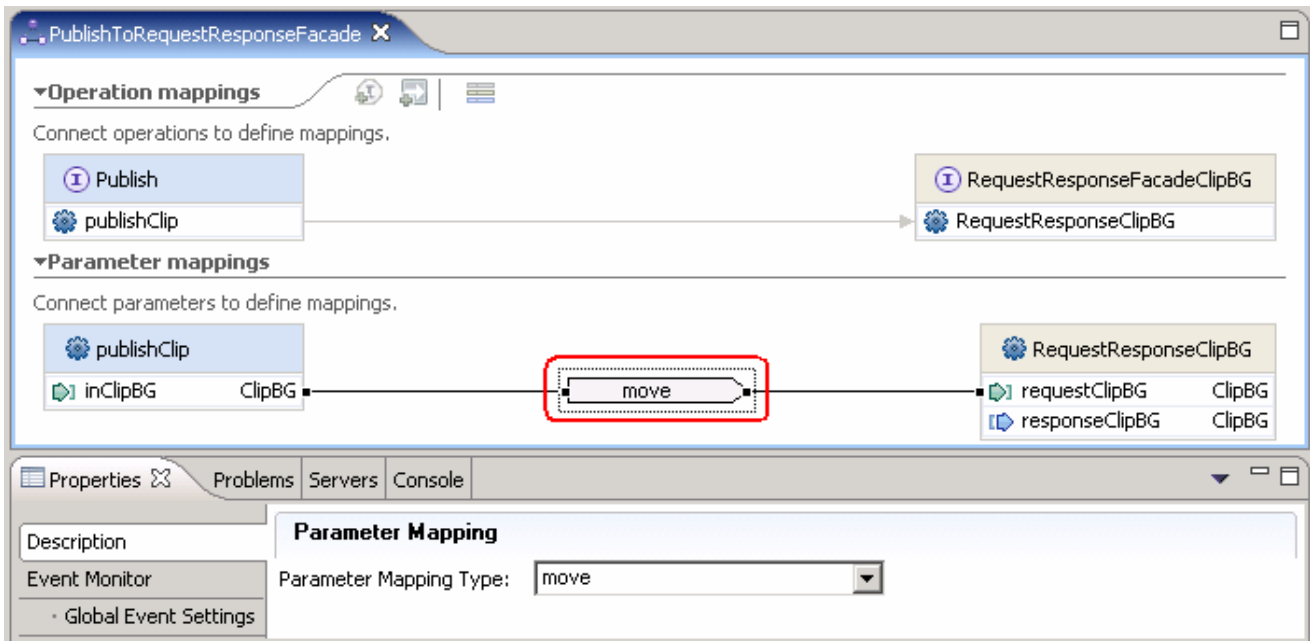
Siebel_ProductTypeRequest (Siebel_ProductTypeBG) → map → requestClipBG (ClipBG)
Siebel_ProductTypeResponse (Siebel_ProductTypeBG) → map → responseClipBG (ClipBG)
Fail → WBIAdapterResponse

Parameter Mapping

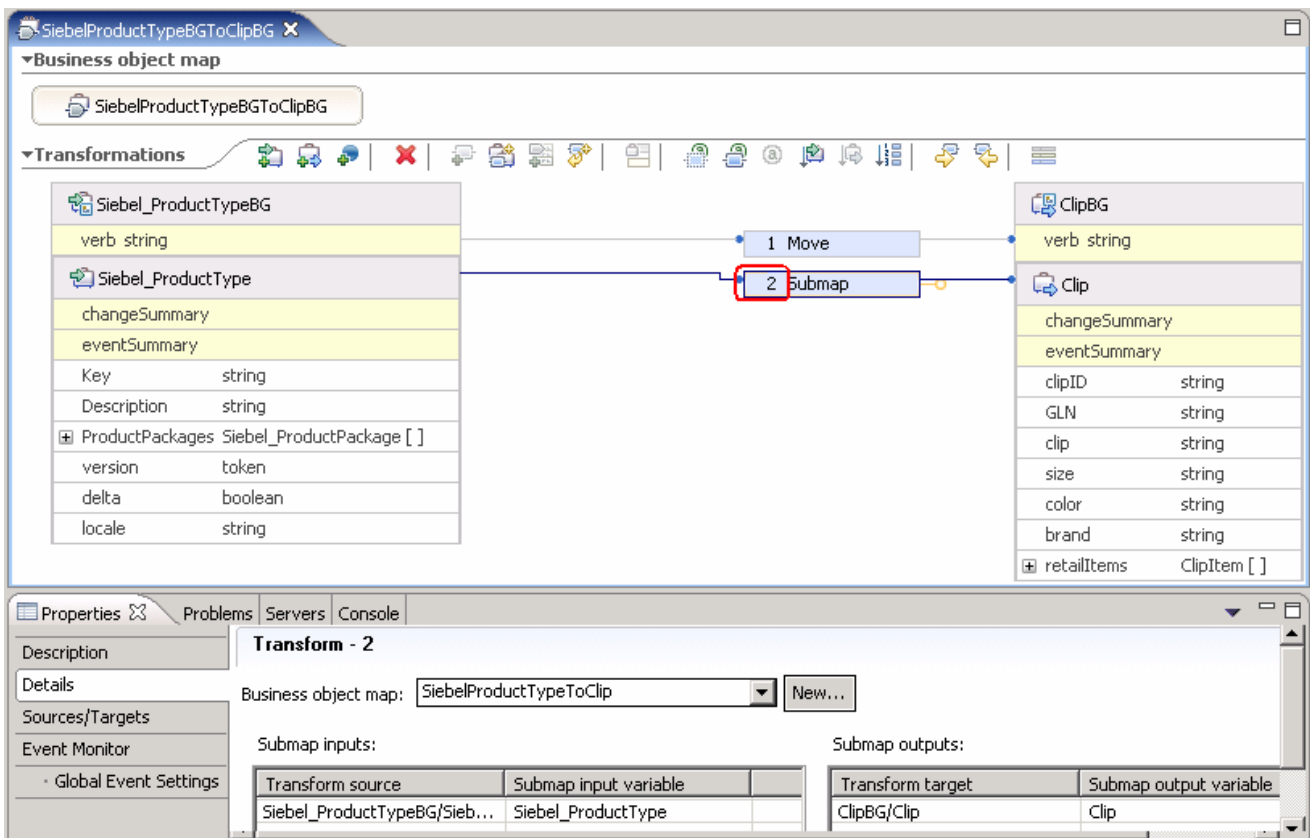
Business Object Map: SiebelProductTypeBGToClipBG

Map input	Input parameter	Output parameter	Map output
Siebel_ProductTypeBG	Siebel_ProductTypeRes...	responseClipBG	ClipBG

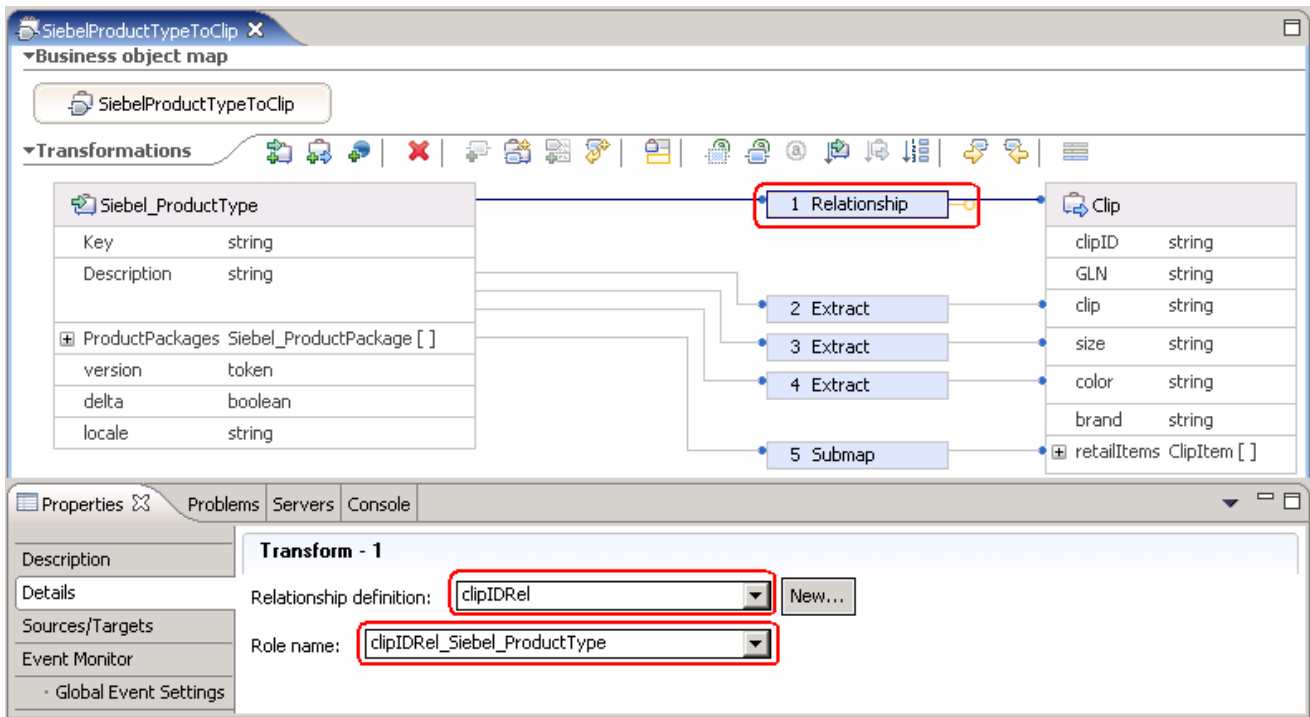
- **PublishToRequestResponseFacade** – to map from the export to the new interface map.



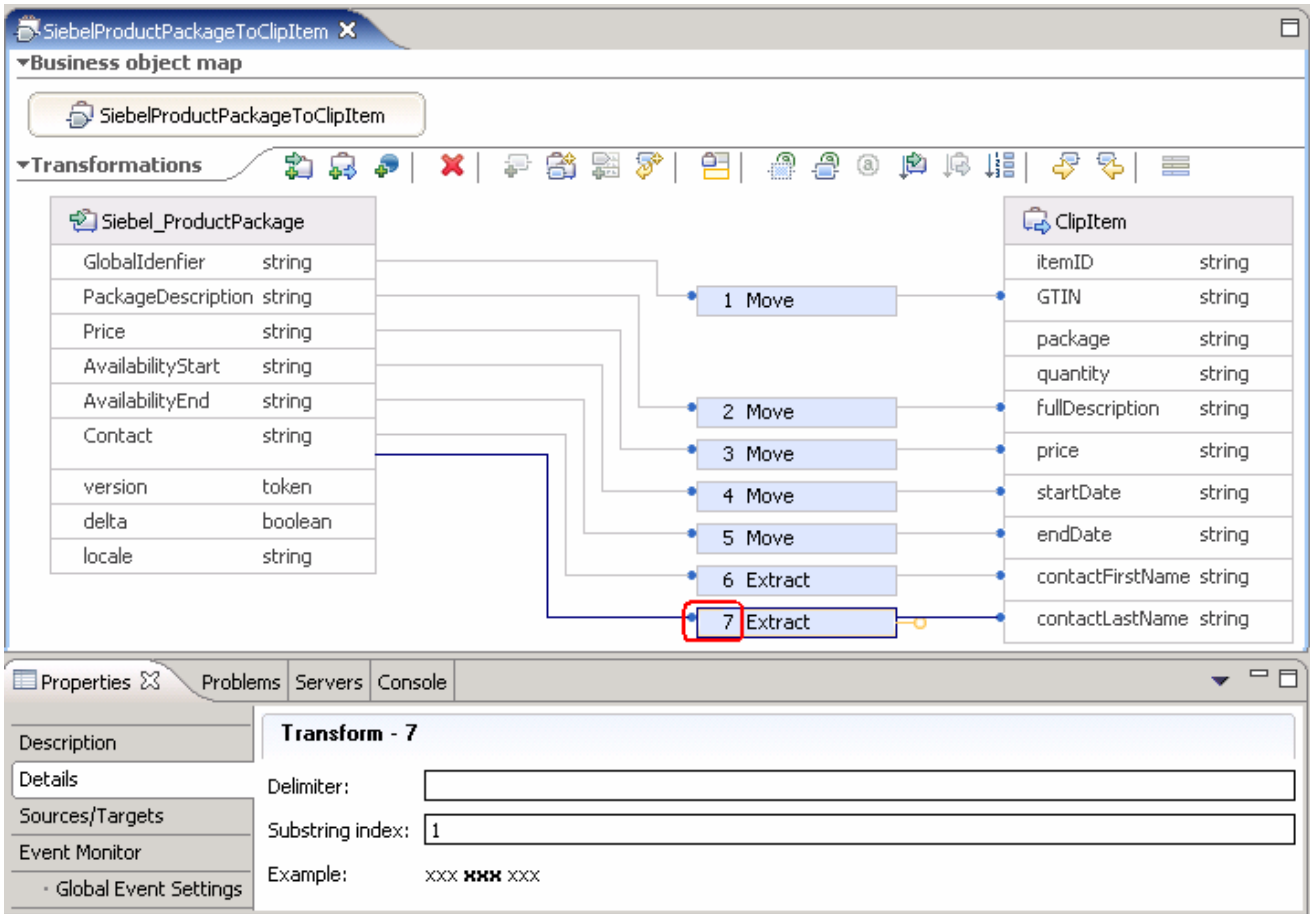
- Data Maps
 - **SiebelProductTypeBGToClipBG** – top-level map for the BG.



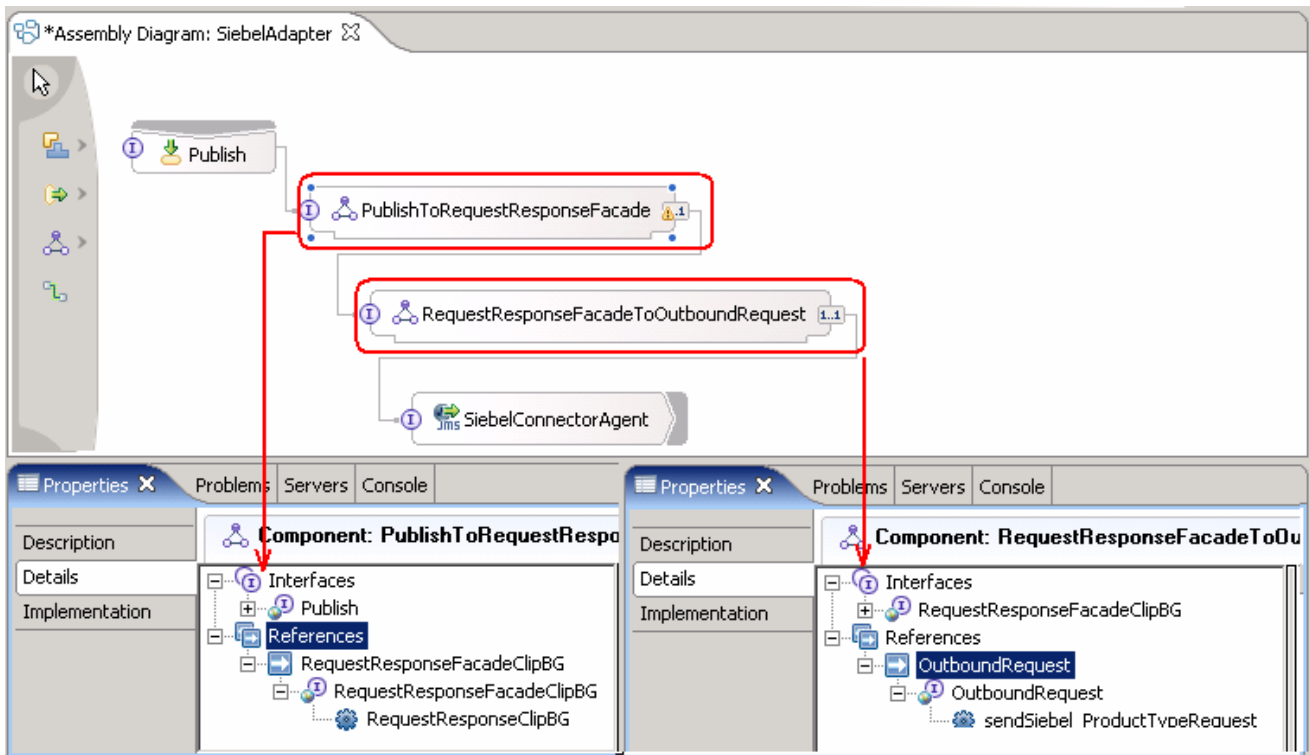
- **SiebelProductTypeToClip** – submap for the parent BO – contains the relationship.



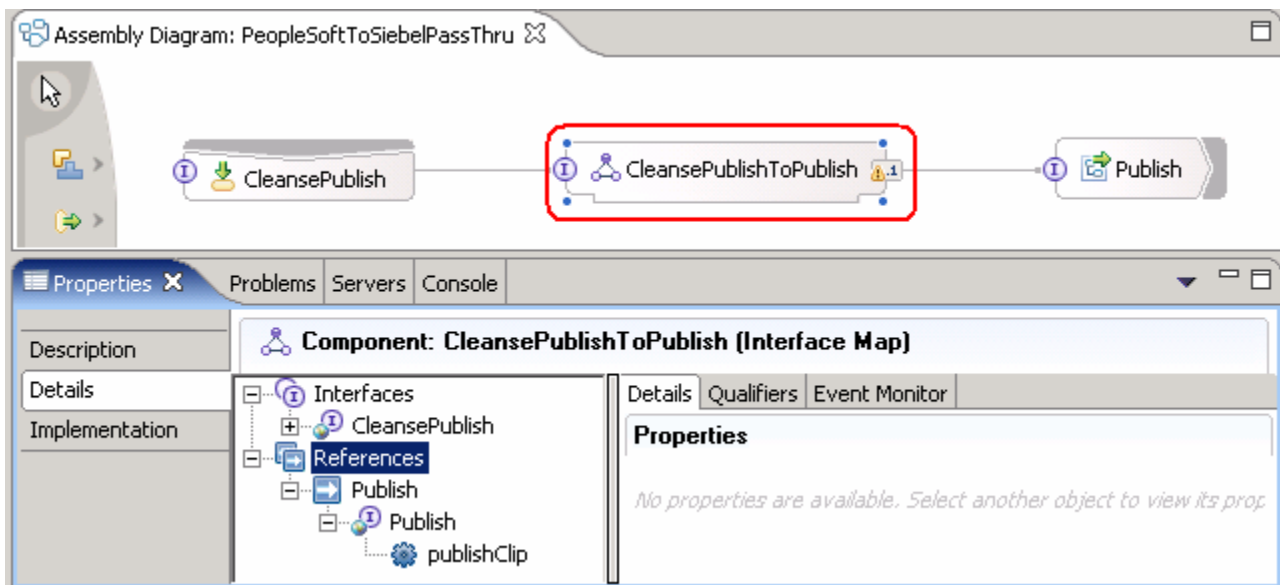
- **SiebelProductPackageToClipItem** – submap for the child BO.



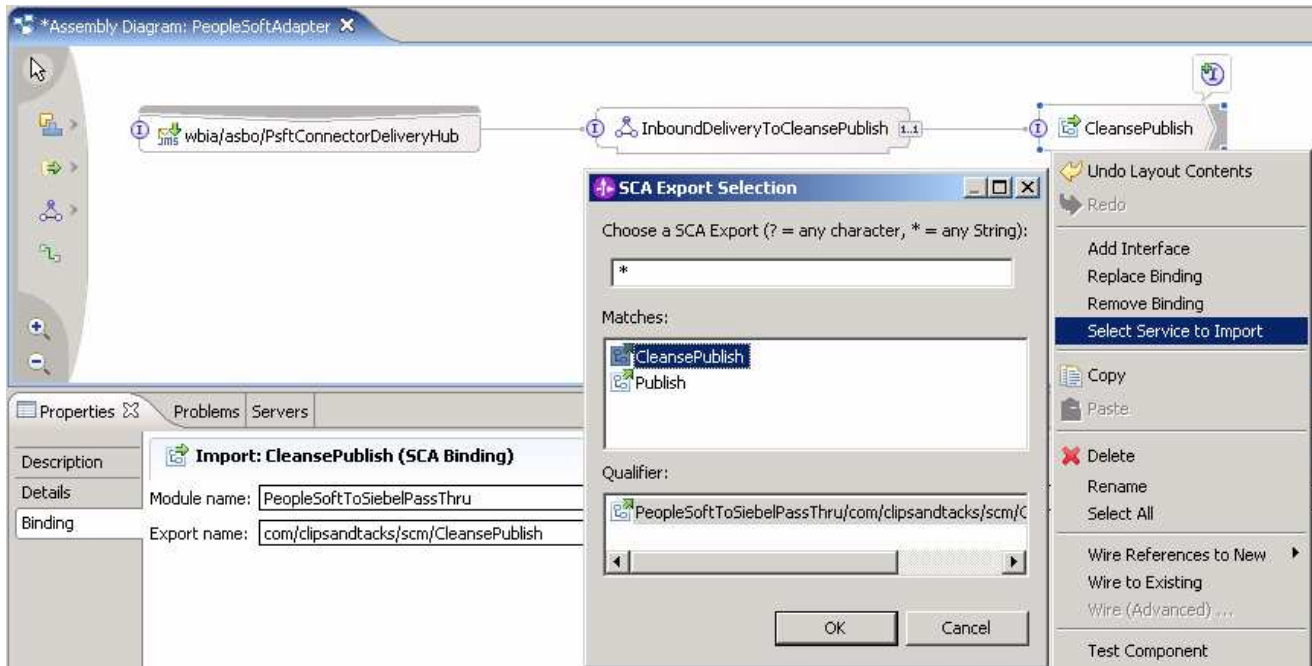
The **SiebelAdapter** module wiring diagram would now look like this:



The new **PeopleSoftToSiebelPassThru** module wiring diagram would look like this:



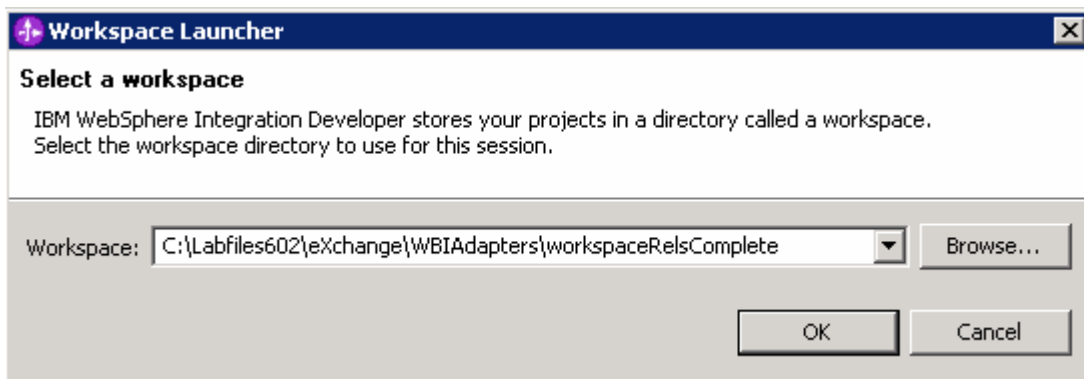
The **PeopleSoftAdapter** module wiring diagram would now look like this:



Part 4: Initialize the workspace for the testing exercise

To complete the exercise, you would need to author everything that was described in Part 3. Rather than having you do that, a Project Interchange file has been provided which contains both of the mappings and relationships for PeopleSoft and Siebel along with the additional interfaces and mediations described in Part 3.

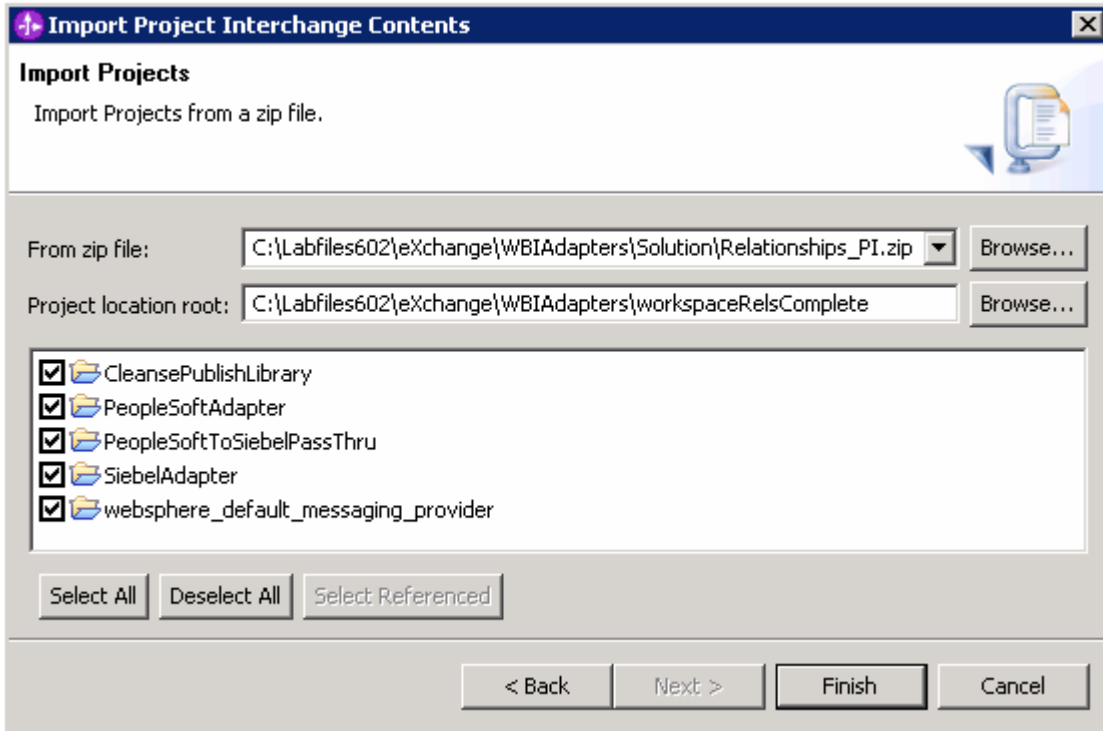
- ___ 1. Switch to a new workspace location of **<WORKSPACE>**, that is **C:\Labfiles602\Exchange\WBIAdapters\workspaceRelsComplete**
 - ___ a. From the main menu, select **File → Switch Workspace...**
 - ___ b. When prompted for workspace, enter the value provided by the **<WORKSPACE>** variable for 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, **Relationships_PI.zip** located at **<LAB_FILES>\Exchange\WBIAdapters\Solution**
 - ___ 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\Relationships_PI.zip** and hit **Open**



__ e. Click **Finish**

___ 3. At this point you should be able to use the editors without instructions to examine the imported artifacts. Review the following information:

__ a. SiebelAdapter assembly diagram (SiebelAdapter → Assembly Diagram)

__ b. RequestResponseFacadeClipBG Interface (CleansePublishLibrary → Interfaces)

__ c. PublishToRequestResponseFacade Interface Map (SiebelAdapter → Mapping → Interface Maps)

__ d. RequestResponseFacadeToOutboundRequest Interface Map (SiebelAdapter → Mapping → Interface Maps)

__ e. SiebelProductTypeBGToClipBG BG Map (SiebelAdapter → Mapping → Data Maps)

__ f. SiebelProductTypeToClip BO Map (Submap) (SiebelAdapter → Mapping → Data Maps)

__ g. SiebelProductPackageToClipItem BO Map (Submap) (SiebelAdapter → Mapping → Data Maps)

___ 4. Add your projects to the server configured projects

__ a. Right click the WebSphere Process Server V6.0 server in the Servers view and select **Add and Remove Projects** from the context menu

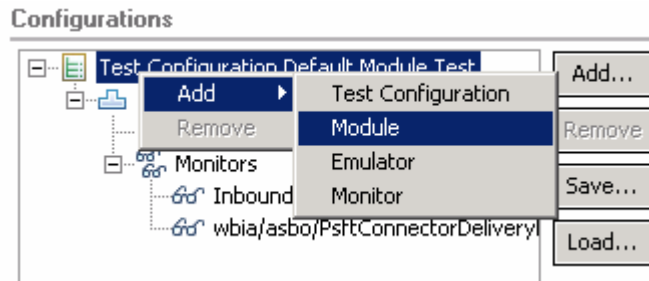
__ b. Click **Add >** to add it to the Configured projects panel

__ c. Click **Finish**

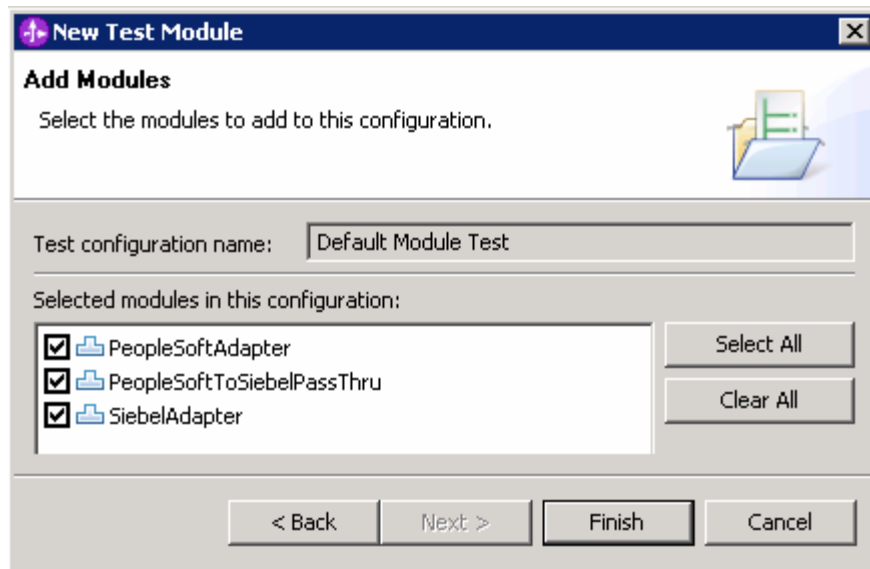
__ d. Wait for the server to publish the projects and to fully start

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) to add a server to the WebSphere Integration Developer test environment and start it. This is especially true for z/OS, AIX, Solaris remote test environment, where the WebSphere Integration Developer will be remote to the test environment

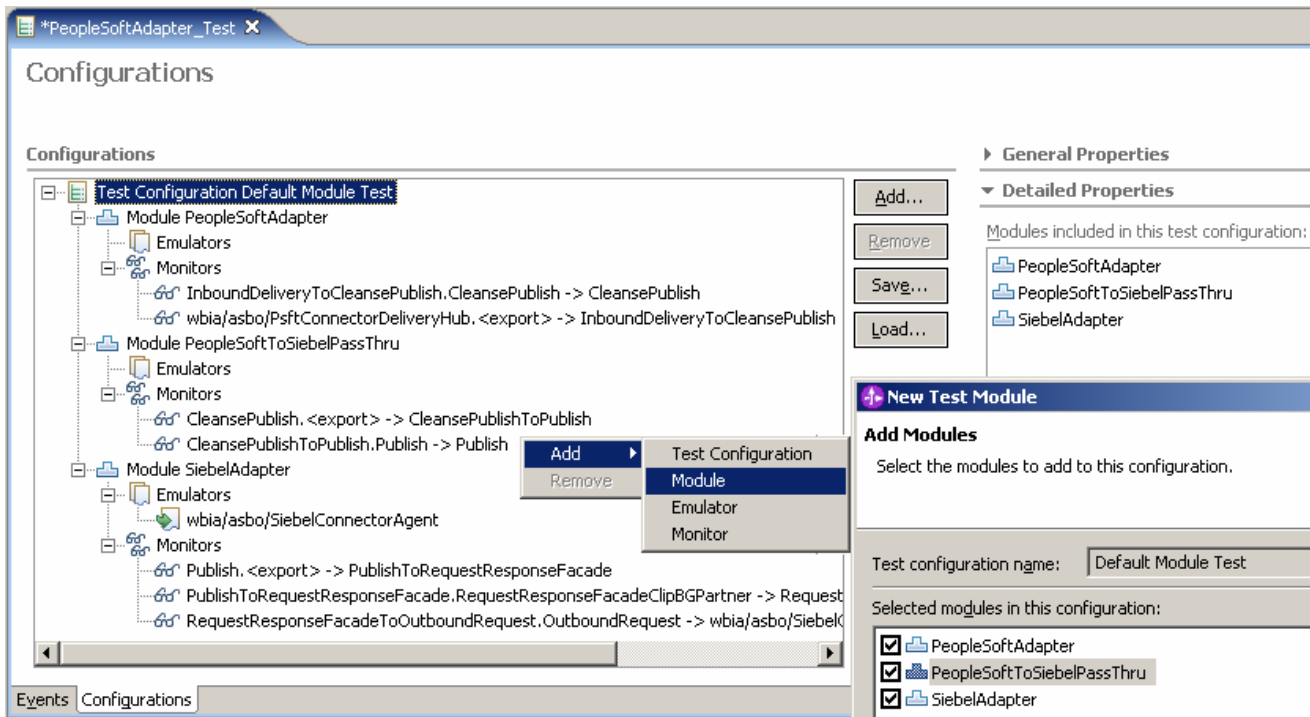
- ___ 5. Configure the Test Component to unit test your projects
- ___ 6. Add Modules to the Test Configuration
 - ___ a. In the Business Integration view, right click the PeopleSoftAdapter module and select **Test → Test Module** from the context menu
 - ___ b. Select the **Configurations** tab at the bottom of the Test Component view
 - ___ c. Right click on the “**Test Configuration Default Module Test**”, that is the top level node and select **Add → Module** from the context menu



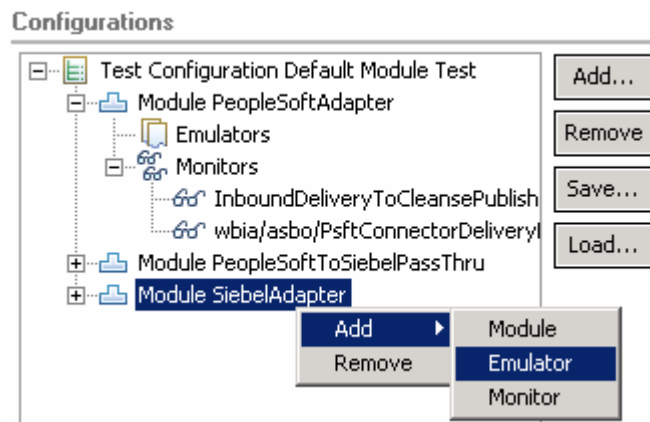
- ___ d. Click **Next** at the initial prompt
- ___ e. Click the **Select All** button to add the missing two modules to this test configuration



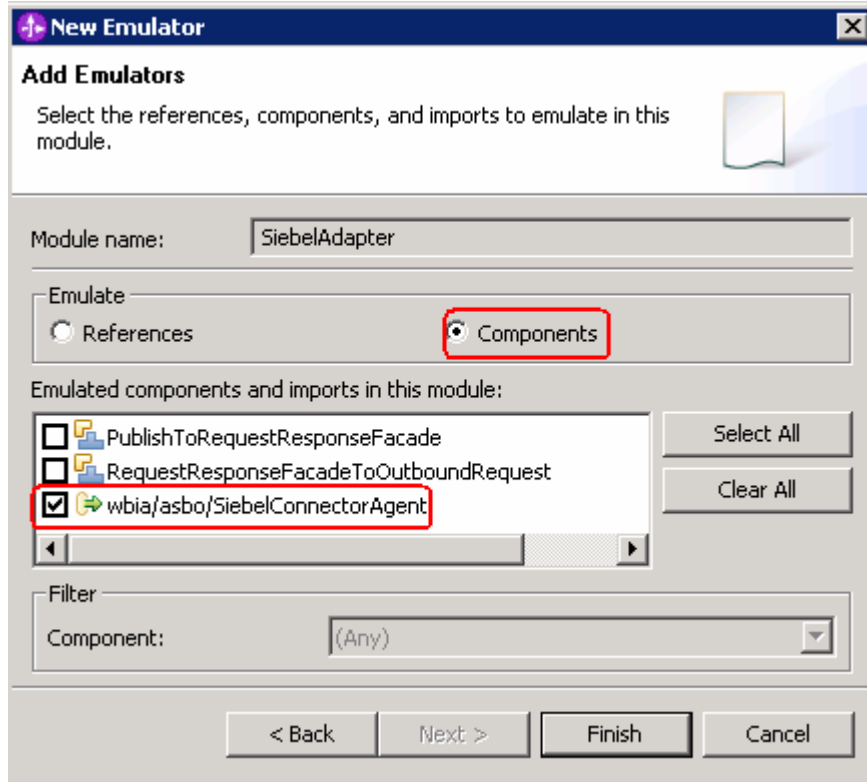
- ___ f. Click **Finish**



- ___ 7. Add an Emulator to SiebelAdapter
 - ___ a. Right click the **Module SiebelAdapter** node and select **Add > Emulator** from the menu



- ___ b. Accept the default and click **Next** at the initial prompt
- ___ c. Select the radio button next to **Components**
- ___ d. Select the check box next to **wbia/asbo/SiebelConnectorAgent** component



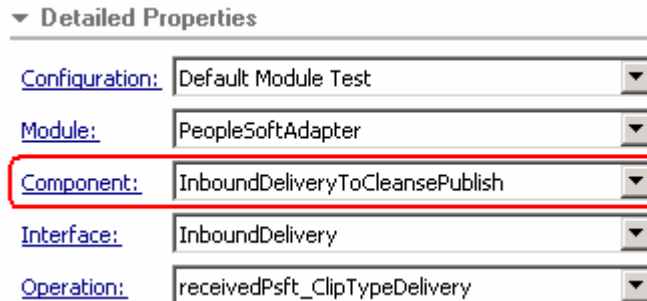
__ e. Click **Finish**

___ 8. Invoke the Test Component and unit test your projects

__ a. Select the **Events** tab at the bottom of the Test Component view

__ b. Make sure that the **Detailed Properties** match with the following screen capture

1) Select **InboundDeliveryToCleansePublish** for the **Component** field using the drop down list capability



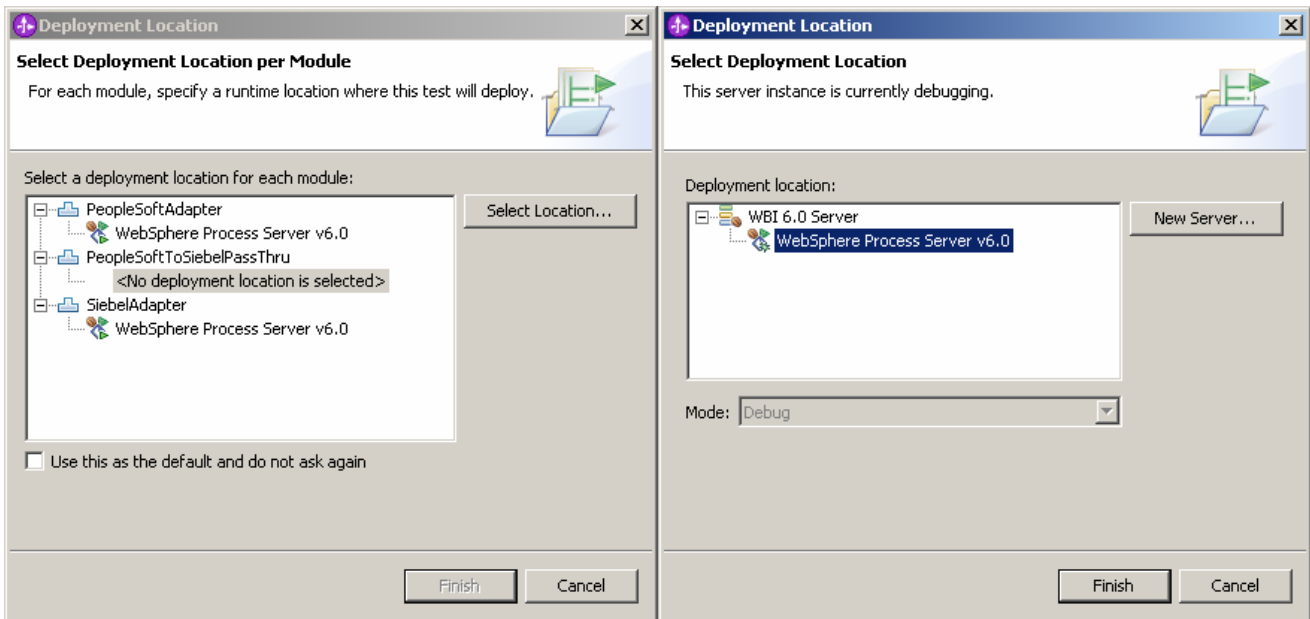
__ c. Provide sample data for the Psft_ClipTypeBGEElement, you must provide the **verb** (such as Create) and a unique **ClipID** value

Initial request parameters

Name	Type	Value
[-] Psft_ClipTypeB... verb	Psft_ClipTypeBG string	Create
[-] Psft_ClipType ClipID	Psft_ClipType string	101
Clip	string	clipp
Size	string	ssize
Color	string	collor
RetailUnits	Psft_RetailItem []	<null>
version	token	
delta	boolean	false
locale	string	

___ d. Click the **Continue** button beneath the Invocation parameters view

___ e. Select each module that does not have the deployment location and then click **Select Location**



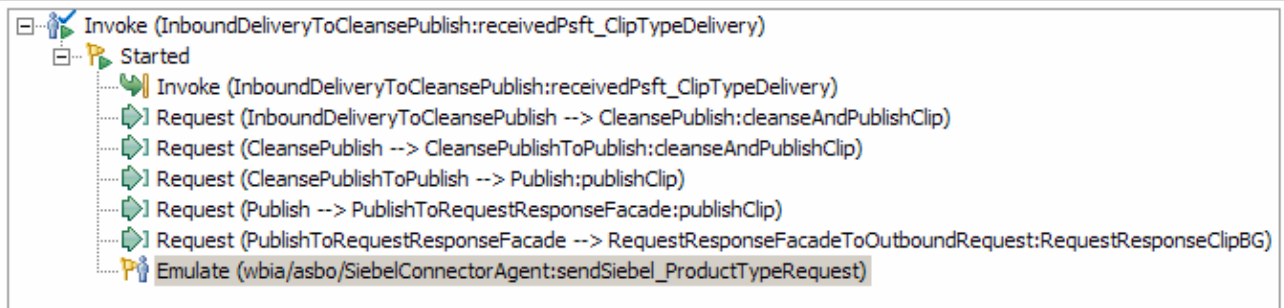
___ f. Select the appropriate WebSphere Process Server on which you want to deploy the application. This is especially true for z/OS, AIX, Solaris remote test environment, where the WebSphere Integration Developer will be remote to the test environment. Click **Finish**

Note: If you get exceptions in the test component window, stop and close the test component and then restart the server with the applications still added to the server. When the server finishes restarting, continue from step 4 in part 4. 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) to restart a server.

___ 9. Observe the Test Component start routine and wait for it to reach the Emulator event

___ a. Examine the already run events once the Test Component pauses

Events



- ___ b. Select the Emulate (wbia/asbo/SiebelConnectorAgent:sendSiebel_ProductTypeRequest) event
- ___ c. Scroll down in the **Detailed Properties** view and locate the **Output parameter** view
- ___ d. You must provide the response data emulating the Siebel Adapter responding to the system. This can be done by doing a copy of **Siebel_ProductTypeBGElement** in the Input Parameters and then doing a paste to **Siebel_ProductTypeBGElement** in the Output Parameters
- ___ e. Ensure you entered the correct **verb** when starting the Test Component (Create)
- ___ f. Enter a unique Siebel **key** value (emulating the Siebel application creating a new entry in its data store). For example, 1001 as shown in the screen capture

The screenshot displays the 'Detailed Properties' panel for a test component. The component is identified as 'SiebelAdapter' within the 'RequestResponseFacadeToOutboundRequest' source component. The target is 'wbia/asbo/SiebelConnectorAgent' implementing the 'OutboundRequest' interface, with the specific operation being 'sendSiebel_ProductTypeRequest'.

The 'Input parameters' table is as follows:

Name	Type	Value
Siebel_ProductT... verb	Siebel_ProductTyp... string	Create
Siebel_Produc... Key	Siebel_ProductType string	<unset>
Description	string	clipp ssize collar
ProductPac... ProductPac...	Siebel_ProductPac... Siebel_ProductPac...	<null>
version	token	0.0.0
delta	boolean	false
locale	string	

The 'Output parameters' table is as follows:

Name	Type	Value
Siebel_ProductT... verb	Siebel_ProductTyp... string	Create
Siebel_Produc... Key	Siebel_ProductType string	1001
Description	string	size color clip
ProductPac... ProductPac...	Siebel_ProductPac... Siebel_ProductPac...	<null>
version	token	
delta	boolean	false
locale	string	

In the output table, the 'Create', '1001', and 'size color clip' values are circled in red.

___ g. Click the **Continue** button and observe the Test Component finishing up the unit test flow

___ h. Select the first **Response** event and verify the response Siebel data sent back to the system

The screenshot shows the 'Events' pane on the left with a tree view of events. The 'Detailed Properties' pane on the right is expanded to show the following information:

- Module: [SiebelAdapter](#)
- Source component: [RequestResponseFacadeToOutboundRequest](#)
- Source reference: [OutboundRequest](#)
- Target component: [wbia/asbo/SiebelConnectorAgent](#)
- Target interface: [OutboundRequest](#)
- Target operation: [sendSiebel_ProductTypeRequest](#)

Response parameters table:

Name	Type	Value
Siebel_ProductT...	Siebel_ProductTyp...	
verb	string	Create
Siebel_Product...	Siebel_ProductType	
Key	string	1001
Description	string	size color clip
ProductPac...	Siebel_ProductPac...	<null>
version	token	
delta	boolean	false
locale	string	

- ___ i. Select the second **Response** event and verify that the response ClipBG object unique key (clipID) matches the unique key value that assigned to the request ClipBG by the RelationshipService

The screenshot shows the 'Events' pane on the left with a tree view of events. The 'Detailed Properties' pane on the right is expanded to show the following information:

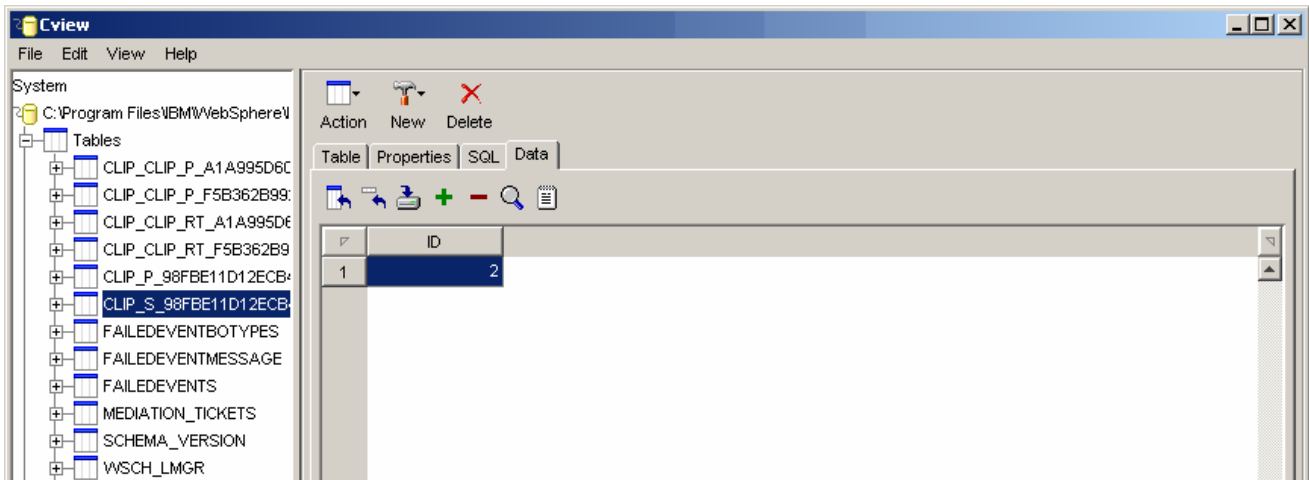
- Module: [SiebelAdapter](#)
- Source component: [PublishToRequestResponseFacade](#)
- Source reference: [RequestResponseFacadeClipBG](#)
- Target component: [RequestResponseFacadeToOutboundRequest](#)
- Target interface: [RequestResponseFacadeClipBG](#)
- Target operation: [RequestResponseClipBG](#)

Response parameters table:

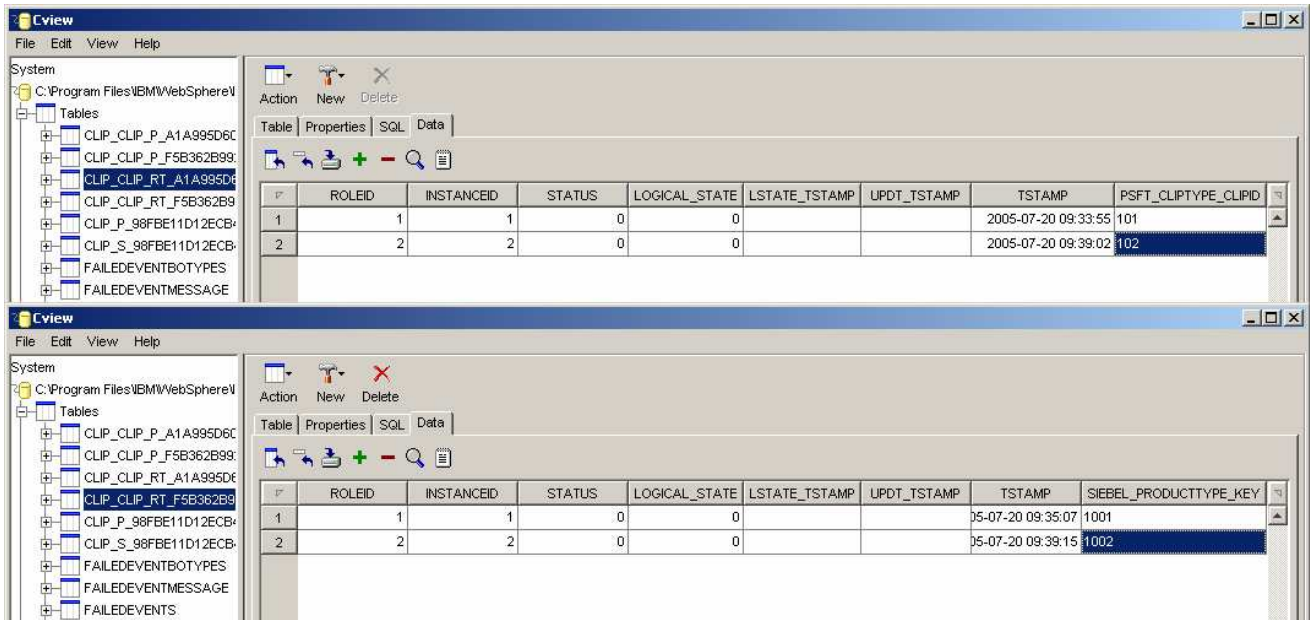
Name	Type	Value
responseClipBG	ClipBG	
verb	string	Create
Clip	Clip	
clipID	string	1
GLN	string	<unset>
clip	string	color
size	string	size
color	string	clip
brand	string	<unset>
retailItems	ClipItem []	<null>

- ___ 10. Rerun the Invocation from within the already configured Test Component several times, using unique values for both PeopleSoft and Siebel BG primary keys each time

- ___ 11. Right click the server and **Remove all** projects from your server instance
- ___ 12. Stop the server before proceeding with the next exercise
 - ___ a. Right click on WebSphere Process Server V6.0 server from the Servers view and select Stop from the context menu
- ___ 13. Examine the Relationship data stored in the WPSDB database (in Cloudscape)
 - ___ a. In the Windows Explorer, navigate to **<WID_HOME>\runtimes\bi_v6\cloudscape\bin\embedded**
 - ___ b. Double-click the **cvview.bat** file to start the **Cloudview** utility to manage Cloudscape databases
 - ___ c. From the main menu, select **File → Open...** and navigate to **<WID_HOME>\runtimes\bi_v6\cloudscape\databases**
 - ___ d. Select the **WPRCSDB** database and click **Open**
 - ___ e. Expand the **Tables** node and examine the list of existing database tables
 - ___ f. Select the **CLIP_S_***** table and then select the **Data** tab in the right view pane. This table stores the sequence counter for the managed clipID relationship role



- ___ g. Select the **CLIP_CLIP_RT_***** tables and examine their content. These tables store the relationship information for the non-managed relationship roles



___ h. From the main menu, select **File** → **Exit** to close the Cloudview utility when done examining the relationship data

What you did in this exercise

You learned how to author identity relationships and how they are associated with data maps. You also learned about the requirements for an identity relationship to be completed during a create flow. Because the original design did not account for this, the design changes required were imported.

Solution instructions

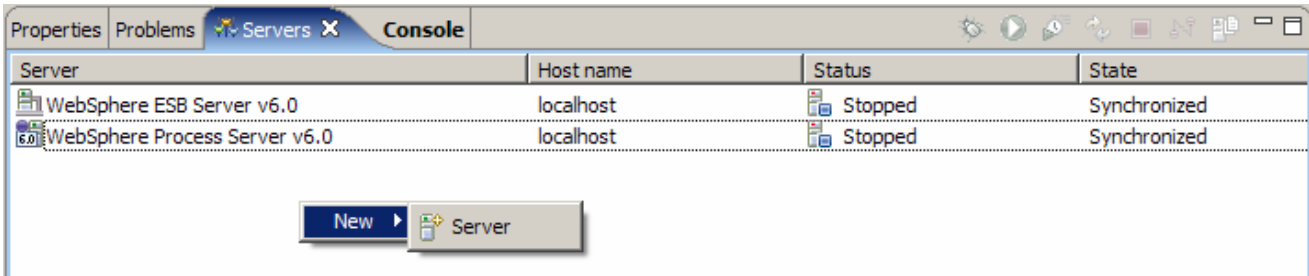
There are no solution instructions for this lab exercise.

Task: Adding remote server to WebSphere Integration Developer test environment

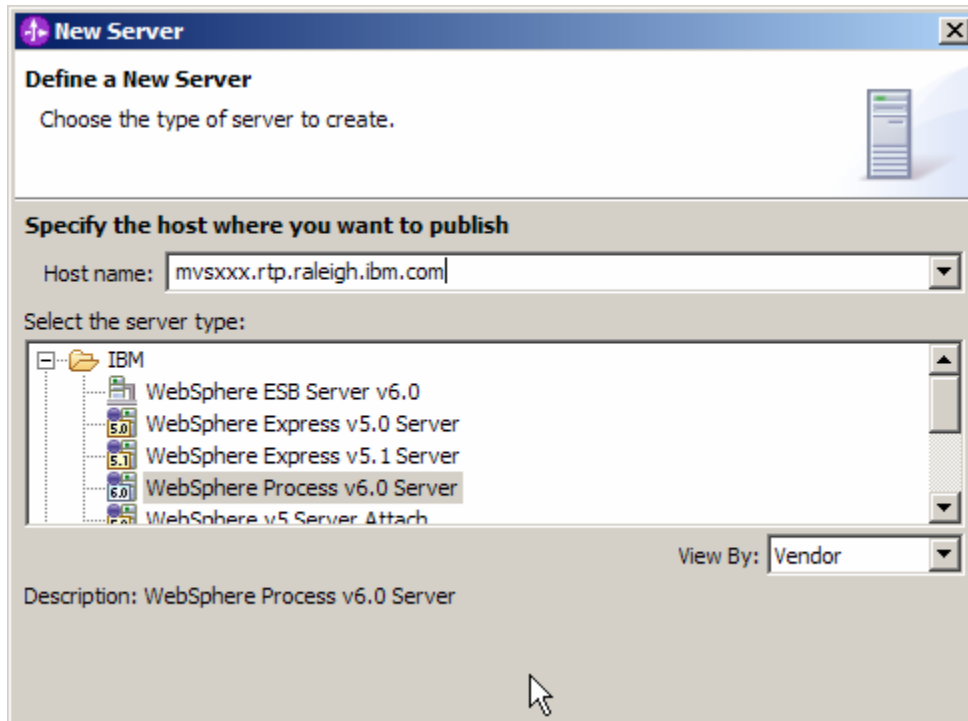
This task describes how to add a remote server to the WebSphere Integration Developer Test environment. The sample you will use is a z/OS machine.

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**.

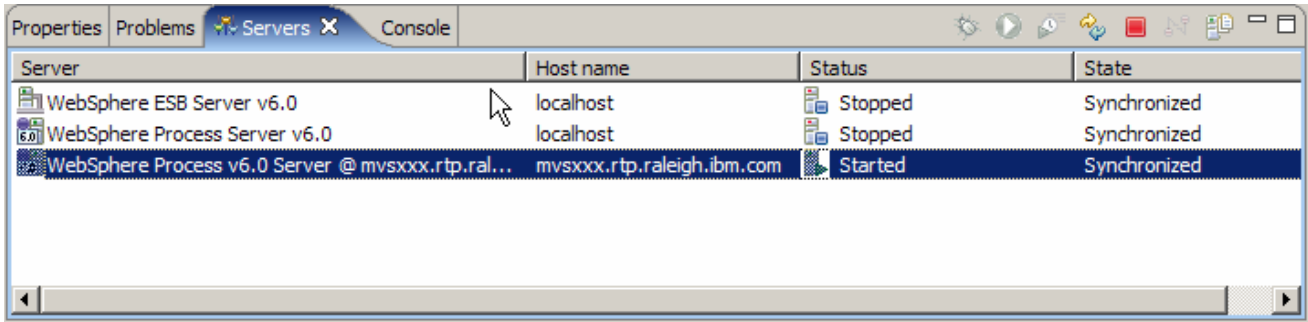
- ___ 6. On the WebSphere Server Settings page, select the radio button for **RMI** and change the ORB bootstrap port to the correct setting (<BOOTSTRAP_PORT>).

The screenshot shows a 'New Server' dialog box with the following settings:

- WebSphere profile name:** [Empty dropdown]
- 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:** [Empty text box]
 - Password:** [Empty text box]
- Server name:** server1
- Server type:**
 - BASE, Express or unmanaged Network Deployment server**
 - Network Deployment server**
 - Network Deployment server name:** [Empty text box]
 - 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.

Buttons at the bottom: < 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
```