#### IBM WEBSPHERE PROCESS INTEGRATION 6.0 – LAB EXERCISE

# Flat Files JCA Adapter Lab

What this exercise is about	. 1
Lab Requirements	1
What you should be able to do	2
Introduction	2
Exercise Instructions	3
Part 1: Initialize the Workspace for this Lab Exercise	4
Part 2: Create the FlatFilesJCAAdapter Application	. 8
Part 3: Test the application using the WebSphere Test Environment (WTE)	25
What you did in this exercise	33

#### What this exercise is about

The objective of this lab is to provide you with an understanding of the IBM WebSphere Adapter for Flat Files and how to deploy and integrate it with an SCA application for outbound processing.

## Lab Requirements

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

- WebSphere Integration Developer V6 installed.
- WebSphere Process Server V6 test environment installed.
- WebSphere Adapter for Flat Files V6 installed.
- Optionally, review the Technotes located at <u>http://www-306.ibm.com/software/integration/wbiadapters/support/</u> for the adapter.
- Interim Fix PK12908 for WebSphere Application Server 6.0.1.2 located at <a href="http://www-1.ibm.com/support/docview.wss?uid=swg24010665">http://www-1.ibm.com/support/docview.wss?uid=swg24010665</a> and referred to by WebSphere Adapter Technote located at <a href="http://www-1.ibm.com/support/docview.wss?rs=695&context=SSMKUK&q1=waffv60rnote&uid=swg21217657&loc=en\_US&cs=utf-8&lang=en">http://www-1.ibm.com/support/docview.wss?rs=695&context=SSMKUK&q1=waffv60rnote&uid=swg21217657&loc=en\_US&cs=utf-8&lang=en</a>

Applying this interim fix resolves the exceptions described in this lab Part 3, step 3., i., in the **Note:** box. The exceptions are benign, without the interim fix; however applying the interim fix resolves the problem so the exceptions are no longer thrown.

• Sample code in the directory C:\Labfiles60 (Windows) or /tmp/LabFiles60 (Linux)

# What you should be able to do

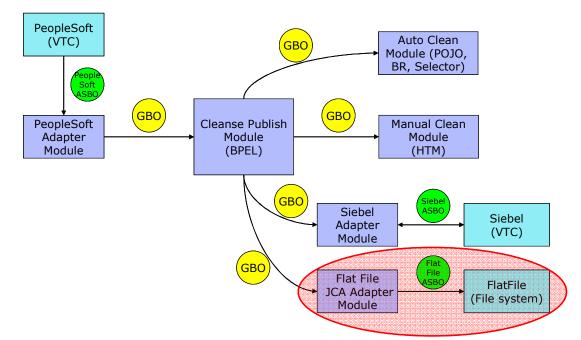
Install and deploy the WebSphere Adapter for Flat Files integrated within an SCA application. Create applications processing outbound requests to the file system using the Flat Files adapter.

## Introduction

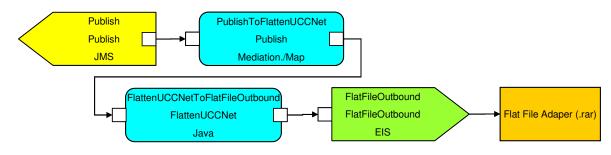
In this lab you will deploy the WebSphere Adapter for Flat Files, using WebSphere Integration Developer, and integrate it into an SCA application that processes outbound requests to the file system. As part of the application, you will create a Java component that uses the Publish Interface, and accepts the Business Objects ClipBG, Clip, and ClipItem, all from the eXchange Lab. The Java component implementation uses the BOXMLSerializer to serialize the objects into bytes which the adapter then sends out in a file to the file system. You will also create an SCA export that supports the Publish Interface, enabling this SCA application to be available to other SCA applications that may want to invoke it.

#### **Overview of Exercise**

The following diagram highlights the parts of the overall scenario that will be addressed in this lab.



# **Description of the Module**



#### **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 execute 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></wid_home>	C:\Program Files\IBM\WebSphere\ID\6.0	/opt/IBM/WebSphere/ID/6.0
<wps_home></wps_home>	C:\ <wid_home>\runtimes\bi_v6</wid_home>	<wid_home>/runtimes/bi_v6</wid_home>
<lab_files></lab_files>	C:\Labfiles60	/tmp/Labfiles60
<workspace></workspace>	C:\Labfiles60\eXchange\FlatFile\workspa ce	/tmp/Labfiles60/eXchange/FlatFile/work space
<ffadapter_h OME&gt;</ffadapter_h 	C:\Program Files\IBM\ResourceAdapters\FlatFiles	/opt/IBM/ResourceAdapters/FlatFiles
<temp></temp>	C:\temp	/tmp

**Windows users**: When directory locations are passed as parameters to a Java program such as EJBdeploy or wsadmin, you must replace the backslashes with forward slashes to follow the Java convention. For example, C:\Labfiles60\ would be replaced by C:\Labfiles60/.

#### Part 1: Initialize the Workspace for this Lab Exercise

1. Follow the directions below to initialize the Workspace using the following values:

#### <WORKSPACE>

C:\Labfiles60\eXchange\FlatFile\workspace

#### <PROJECT\_INTERCHANGE>

C:\Labfiles60\eXchange\CleansePublishLibrary\import\CleansePublishLibrary\_PI.zip

#### <MODULE>

FlatFilesJCAAdapter

#### <DEPENDENT\_LIBRARIES>

CleansePublishLibrary

Note: A subdirectory named C:\Labfiles60\eXchange\FlatFile\outdir has been created for you. This is the specified subdirectory in which the application creates a file named *publish.txt* as part of the exercise. If you have previously run any lab with the Flat File Adapter that created a file named *publish.txt* in the outdir subdirectory (for example, the "End to End Scenario" lab, delete that file(s) before continuing; otherwise, you will get an exception later in this lab when trying to create the *publish.txt*, that the file already exists.

- \_\_\_\_2. Start WebSphere Integration Developer V6 with a new workspace located at < WORKSPACE>.
  - \_\_\_\_a. From Windows Explorer, navigate to the <WID\_HOME> directory and double click on wid.exe.
  - \_\_\_\_b. When prompted for workspace name, enter the value provided by the **<WORKSPACE>** variable for this lab and click **OK.**

Workspace Launcher	<
Select a workspace	
Eclipse Platform stores your projects in a directory called a workspace. Select the workspace directory to use for this session.	
Workspace: C:\Labfiles60\eXchange\sample\workspace   Browse	]
☐ Use this as the default and do not ask again	
OK Cancel	

\_\_\_\_ c. When WebSphere Integration Developer V6.0 opens, close the **Welcome page** by clicking on the Go to the workbench icon (bent over arrow at top-right).

🧧 Resource - Eclipse Platform	×
<u>File E</u> dit <u>N</u> avigate Se <u>a</u> rch <u>P</u> roject <u>R</u> un Performance <u>W</u> indow <u>H</u> elp	
🖶 Welcome 🗙 👔 😓 🕁 🗖	8
Welcome to Eclipse 3.0	)
Ma. K	ここう言語

- \_\_\_\_d. Ensure you are in the **Business Integration** perspective.
- 3. If this lab requires you to import a project interchange file, setup the required libraries and modules for this lab by importing the project interchange file **<PROJECT\_INTERCHANGE>**.
  - \_\_\_\_a. Select File -> Import... from the menu bar.
  - \_\_\_\_b. In the Import dialog, scroll down and select Project Interchange.

🔙 Import				X
Select Import a project and its d	ependent projec	ts from a Zip file.		
Select an import source:				
<ul> <li>Log File</li> <li>Portal</li> <li>Probe</li> <li>Profiling file</li> <li>Project Interchange</li> <li>RAR file</li> <li>Security Certificate</li> </ul>				
	< <u>B</u> ack	<u>N</u> ext >	Einish	Cancel

\_\_\_ c. Click Next.

\_\_\_\_d. In the Import Projects dialog, initialize the From zip file: field to <**PROJECT\_INTERCHANGE>.** 

\_\_\_\_e. Click the Select All button.

🔙 Import Project	Interchange Contents	×
Import Projects Import Projects from	a zip file.	
From zip file: Project location root:	C:\Labfiles60\eXchange\CleansePublishLibrary\import\CleansePublishLibrary_PI.zip C:\Labfiles60\eXchange\sample\workspace	Browse
Select All Deselect	CE All Select Referenced	
	< <u>B</u> ack <u>N</u> ext > <u>Finish</u>	Cancel

- \_\_\_f. Click Finish.
- \_\_\_\_4. If this lab requires that you create a Business Integration module called **<MODULE>**, complete these steps.

\_\_\_\_a. Right click on the background of the Business Integration view to access the pop-up menu.

\_\_\_\_b. Select New > Module.

<del>នៃ:</del> Busine	ess Integration 🗙			- 8	
		₿	B		
±1	leansePublishLibrary				
	New		Þ	📑 Projec	:t
	Open Dependency Editor Show Files			📂 Modul	e
	Сору			😭 Librar	у

\_\_\_\_ c. In the New Module dialog, enter **<MODULE>** for the Module Name.

Kew Module
Module Create a new business integration module. A module is a project that is used for development, version management, organizing resources, and deploying to the
runtime environment. Create a new business integration module. A module is a project that i
Module Name       SampleModule         Module Location       Image: Sample Sampl
< Back Next > Einish Cancel

\_\_\_\_d. Click Finish.

5. If this lab requires that **<MODULE>** needs any **<DEPENDENT\_LIBRARIES>**, complete these steps.

- \_\_\_\_a. In the Business Integration view, right click on the **<MODULE>** you just created to access the pop-up menu.
- \_\_\_\_b. Select Open Dependency Editor.



- \_\_\_\_ c. Click the **Add...** button In the Dependency Editor.
- \_\_\_\_d. In the Library Selection dialog, select from the list the <DEPENDENT\_LIBRARIES>.
- \_\_\_e. Click OK.
- \_\_\_\_f. Press Ctrl+S to save the dependencies for this module.

## Part 2: Create the FlatFilesJCAAdapter Application

In this part you will run Enterprise Service Discovery to import the WebSphere Adapter for Flat Files, input the property values to build the Service Description (used to create the Managed Connection Factory), create the necessary SCA artifacts, and assemble them into an SCA application. Since the Flat Files Adapter has a single, pre-defined business object structure, there is no Enterprise Service Discovery for "object discovery" of business objects.

1. Start the Enterprise Service Discovery process and import the Flat Files Adapter Resource Archive (RAR) file. This will create a J2EE Connector Project.

\_\_\_\_a. From the top Menu bar, select File > New > Enterprise Service Discovery.

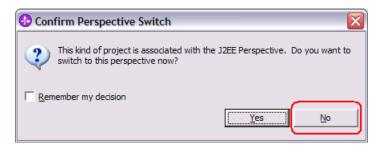
\_\_\_\_b. Click **Import Resource Adapter...** at the bottom right of the Select an Enterprise Service Resource Adapter panel.

Enterprise Service Discovery				
Select an Enterprise Service Resource Ac	lapter			
Select a resource adapter to use to discover a	service.			
WBI Adapter Artifact Importer				
			Import Possa	urce Adapter
Artifact importer for a WBI Adapter endpoint				
	< <u>B</u> ack	<u>N</u> ext >	Einish	Cancel

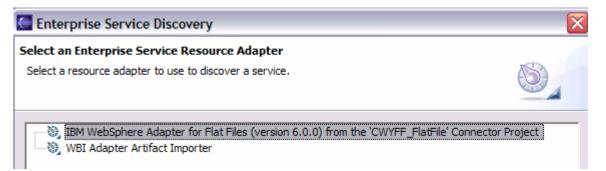
- \_\_\_\_c. Complete the Connector Import panel.
  - 1) Connector file: **Browse...** to the location of the CWYFF\_FlatFile.rar adapter file and select it, for example: <FFADAPTER\_HOME>\adapter\flatfile\deploy\CWYFF\_FlatFile.rar.
  - 2) Deselect the "Add module to an EAR project" if it is selected, and click Finish.

🕑 Import		
Connector Import		
Import a Connector	r project from the file system	
Connector file:	eAdapters\FlatFiles\adapter\flatfile\deploy\CWYFF_FlatFile.rar	Browse
Connector project:	CWYFF_FlatFile	N <u>e</u> w
🔲 Overwrite exis	sting resources without warning,	
🗖 Delete proje	ect on overwrite	
Target server:	WebSphere Process Server v6.0	N <u>e</u> w
	Add module to an EAR project.	
EAR project:	CWYFF_FlatFileEAR	Ne <u>w</u> ,
		1
	Einish	Cancel

\_\_\_\_d. At the Confirm Perspective Switch prompt, select No.



\_ e. You will return to the Select an Enterprise Service Resource Adapter panel. Highlight the **IBM** WebSphere Adapter for Flat Files and click Next.



\_\_\_\_f. From the Configure Settings for Discovery Agent panel, ensure the **ServiceType:** is set to **Outbound**. Click the **Show Advanced** >> button to note the Log file output location and Logging Level options for discovery log, leave the default values, and then click **Next**.

figure Settings for Di	scovery Agent	
ecify the properties to in	itialize the resource adapter and the enterprise service discovery agent.	
		9
onnection Configuration		
Prefix:	* FlatFile	
ServiceType:	Outbound	
🗖 LocalMode		
BiDi Properties		
🧮 BiDi Transforma	ation	
BiDi OrderingSchen	na: Implicit	
BiDi Direction:	LTR	
🔽 BiDi Symmetric	Swapping	
BiDi Shaping:	Initial	
BiDi NumericShapin	g; Nominal	
ide Advanced <<		
gging options		
Log file output location	* C:\Labfiles60\eXchange\workspace\ffadapter\.metadata\FlatFileMetadataDiscoveryImpl.log	Browse
Logging Level:	SEVERE	7

\_\_\_\_g. From the Find and Discover Enterprise Services panel, click the **Run Query** button. An **Outbound** folder will appear in the Objects discovered by query: window. **Select** the **Outbound Folder, click** the >> **Add** button. The **Outbound** folder is now displayed in the Objects to be imported: window. Click **Next.** 

Run Query         Objects discovered by query:         Image: Comparison of the second discovered by query in the second discovered disco			
Use "Edit Query" to create a query and press "Run Query" to discover matching objects on the enterprise system.          Query:       Ed         Run Query       Objects discovered by query:         Discovered by query:       Image: Control of the con		prise Service Discovery	🧲 Er
Run Query         Objects discovered by query:	EIS		
Objects discovered by query:	idit Query		Que
Objects to be imported:			_
Objects to be imported:	>> Add	Dutbound	+
	Filter Clear Filter		
	<< Remove		
	<< Relilove		
< <u>Back</u> <u>N</u> ext > Einish	Cancel	< <u>Back</u> <u>N</u> ext > <u>Finish</u>	

\_\_\_\_h. On the Configure Objects panel, **enter** a BO Location: of **com/test/flat**. Note the Operations available for the ServiceType: Outbound and click **Next**.

Enterprise Service Discovery		X
Configure Objects		
Specify the properties for the objects that will be imported by the discovery	agent.	
ServiceType: Outbound		
NameSpace: http://www.ibm.com/xmlns/prod/websphere/j2ca/flatfile		
BO Location: com/test/flat		
Operations:		
CREATE APPEND		Add
RETRIEVE DELETE		Remove
OVERWRITE EXISTS LIST		
	< <u>B</u> ack <u>N</u> ext >	Einish Cancel

- \_\_\_\_i. On the Generate Artifacts panel, specify the properties for the artifacts that will be generated in the workspace. Leave the default values except as follows:
  - 1) Create a New Module named **FlatFilesJCAAdapter**. (Note: Be sure to type the case as shown. Later you will copy and paste sample Java code that refers to this name, and it is case sensitive.)

Module
Create a new business integration module. A module is a project that is used for development, version management, organizing resources, and deploying to the runtime environment.
Module Name FlatFilesJCAAdapter
Module Location
Directory; c:\Labfiles60\JDBC\betaworkspace2\FlatFilesJCAAda Browse

- 2) Folder: Enter com/test/flat.
- 3) **Select** the radio button to the left of **Use discovered connection properties**. Additional Connection properties and Resource Adapter properties will become visible.
- 4) OutputDirectory: Enter C:\Labfiles60\eXchange\FlatFile\outdir.
  - a) Ensure the C:\Labfiles60\eXchange\FlatFile\outdir exists. If the output directory does not exist, the test will fail. (For this lab, use this specific subdirectory name, however, in your own installations, this value can be whatever you want.)
- 5) OutputFileName: Enter publish.txt.

Deploy connector with module			
Specify the connection properties	Specify the connection properties which will be used to connect to the Enterprise Information System at runtime:		
C Use connection properties	specified on server		
Use discovered connection	n properties		
J2C Authentication Data Entry:			
Connection Properties			
OutputDirectory:	c:\Labfiles60\eXchange\FlatFile\outdir		
OutputFileName:	publish.txt		

\_\_\_ j. Click Finish.

**Note:** The IBM WebSphere Adapters (this includes the IBM WebSphere Adapter for Flat Files, IBM WebSphere Adapter for JDBC, IBM WebSphere Adapter for PeopleSoft Enterprise, IBM WebSphere Adapter for Siebel Business Applications, and IBM WebSphere Adapter for SAP Applications) are supported as "Deploy connector with module" only, meaning the adapter is deployed within the module which is packaged as an Enterprise Archive file (EAR file). Therefore, the "Deploy connector with module" checkbox should always be checked and the "Use discovered connection properties" checkbox should always be checked.

2. You will return to the Business Integration view. While not necessary, if you would like to view the Connector Project, examine the RAR and the deployment descriptor ra.xml file, switch to the Resource view. From the top level menu select Window > Open Perspective > Other, then select Resource. Click OK.

Select Perspective
Business Integration (default) Debug I2EE Java Java Browsing Java Type Hierarchy Resource
Show all
OK Cancel

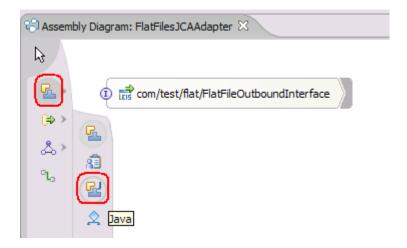
\_\_\_\_a. Expand the CWYFF\_FlatFile folders. Under the META-INF folder, Right click the ra.xml file, and Open with > Text Editor.

🔁 Navigator 🛛			
🕀 😥 CleansePublishLibr	ary		
🖃 避 CWYFF_FlatFile			
🚊 🗁 connectorMod	ule		
🖻 🗁 META-INF			
📄 discov	ery-service.xml		
	EST.MF		
🛅 ra.xml	New	E.	
CWYBS_A		·	
CWYFF_FI	Open		
S FlatFile.xs	One on Mitch	•	🛅 JAR Dependency Editor
icu4j_3_2.	-		Web Browser
.classpath	Copy		_
.j2ee	💼 Paste		XML Source Page Editor
.project	💢 Delete		
.runtime	•••		Text Editor
	Move		- · ·

- \_\_\_\_b. Note the various properties and their values. You will see that the inbound properties are not set, they are not needed as this is a ServiceType of Outbound. Be careful to not change anything. Close the ra.xml file, and return to the Business Integration view. Select **Window** > **Open Perspective** > **Other** > **Business Integration** and click **OK**.
- \_ 3. Use the Assembly Diagram to wire a Java Component to the **FlatFileOutboundInterface**.
  - \_\_\_\_a. Expand the **FlatFilesJCAAdapter** folder from the **Business Integration** perspective.
  - \_\_\_\_b. Double-click the **FlatFilesJCAAdapter** module. This will open the module in the Assembly Diagram. You will see a message that there is one new element added to the module.

😫 Assemb	oly Diagram: FFOutbound 🖾		
Ø		(1) There are 1 new element(s) that have been added to your module.	23
<b>P</b> . >	1 🖢 com/test/flat/FlatFileOutboundInterface		
<b>&gt;</b>			
_&>			

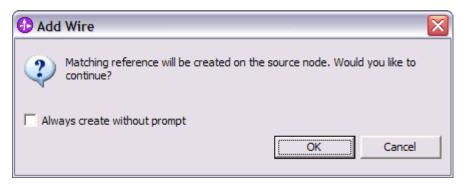
\_\_\_ c. From the palette, select the Component (with no implementation type) icon, then select the Java Component and click in the Assembly Diagram to drop the component on to the Assembly Diagram editor.



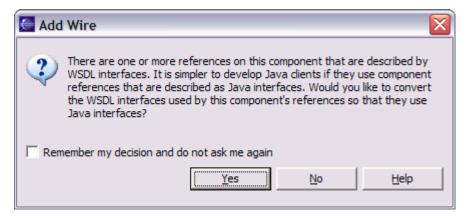
\_\_\_\_d. Wire the Java Component to the com/test/flat/FlatFileOutboundInterface.

🖏 *Asser	mbly Diagram: FlatFilesJCAAdapter 🗴
R	
	Component1
(⇔>	<b>9</b>
<u>م</u> ې	
°L,	
	■ ① 📑 FlatFileOutboundInterface

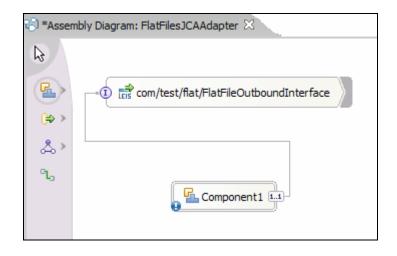
\_ e. Click **OK** on the Add Wire popup window.



\_\_\_\_\_f. If another Add Wire popup window appears as the following, Select No.



\_g. The Assembly Diagram should now look as follows.



- \_\_\_\_\_4. Add an Interface to the Java Component1.
  - \_\_\_\_a. Right click the Java Component1, select Add Interface from the context menu.

	nent1 1.1	com	n/test/flat/FlatFileOutboundInterface	$\rangle$
•	💛 Undo Add Implementation			
	🗣 Redo			
(				
l l	Add		Interface	
	Change Type	•	Reference	
	Export	- <b>F</b>		
	Recenerate Implementation			

\_\_\_\_b. In the Add Interface panel, select the radio button Show WSDL in the middle of the panel, to limit the list, then double click the Publish interface.

🚝 Add Interface	- 🗆 🗙
Filter by interface or qualifier (? = any character, * =	any String):
*	New
Matching interfaces:	
CleansePublish     IFlatFileOutboundInterface	
Publish	
C Show WSDL and Java	
Show WSDL	
C Show Java	
Qualifier:	
①http://CleansePublishLibrary/com/clipsandtacks/s	cm/Publish -
<	>

- \_\_\_\_\_ 5. Generate a Java Implementation for the **Component1Impl.java**.
  - \_\_\_\_a. Right click the **Component1** component, select **Generate Implementation**.

-•① 🕲 com/test/flat/FlatF	ileOutboundInterface
t	
🛈 🛃 Cor	💛 Undo Add Interface
•	😂 Redo
	🗙 Delete
	Сору
	🔁 Paste
	Select All
	Change Type
	Export •
	Generate Implementation
	Merge Implementat

\_\_\_\_b. Highlight the com.test.flat package. Click OK.

Generate Implementation
Please choose the package where the Java implementation will be generated:
New Package OK Cancel

6. Click **No** on the Generate Implementation popup window.

🧲 Gen	erate Implementation
?	There are one or more references on this component that are described by WSDL interfaces. It is simpler to develop Java clients if they use component references that are described as Java interfaces. Would you like to convert the WSDL interfaces used by this component's references so that they use Java interfaces?
🗌 Rem	ember my decision and do not ask me again
	<u>Y</u> es <u>N</u> o <u>H</u> elp

\_\_\_\_a. The Java Editor will open with the **Component1Impl.java** file.

**NOTE:** For your convenience, the following code snippets can be found in <LAB\_FILES>\eXchange\FlatFile\snippets\component1impl.txt.

\_\_\_\_ b. Add the following lines of imports directly below the existing set of import statements at the top of the file.

```
import java.io.ByteArrayOutputStream;
import java.io.ObjectOutputStream;
import com.ibm.websphere.bo.BOFactory;
import com.ibm.websphere.bo.BOXMLSerializer;
```

\_\_\_\_ c. Scroll down to the method **public void publishClip(Dataobject inClipBG)** that needs to be implemented and enter the following code. (The code continues on the following page.)

**Note** these two specific set methods in the following code. These values will overwrite any previous values entered during Enterprise Service Discovery. These dataObject.set methods are not required, as the values entered during Enterprise Service Discovery would suffice. This is just an additional example showing how to set these values in code.

dataObject.set("directoryPath", "C:\\Labfiles60\\eXchange\\FlatFile\\outdir"); dataObject.set("fileName", "publish.txt");

try{

ByteArrayOutputStream bos=new ByteArrayOutputStream(); //ObjectOutputStream oos=new ObjectOutputStream(bos);

BOXMLSerializer

xmlser=(BOXMLSerializer)ServiceManager.INSTANCE.locateService("com/ibm/websphere/bo/B OXMLSerializer");

xmlser.writeDataObject(inClipBG.getDataObject("Clip"),"http://CleansePublishLibrary/com/clip sandtacks/scm","Clip",bos);

//invoke the FlatFilesJCAAdapter outbound test
//
byte [] bytearray=bos.toByteArray();

Service service =

(Service)ServiceManager.INSTANCE.locateService("FlatFileOutboundInterfacePartner"); BOFactory boFactory

=(BOFactory)ServiceManager.INSTANCE.locateService("com/ibm/websphere/bo/BOFactory");

- // Create the wrapper element
  - DataObject element=boFactory.createByElement

("http://FlatFilesJCAAdapter/com/test/flat/FlatFileOutboundInterface","create"); DataObject dataObjectx=element.createDataObject("createInput"); DataObject dataObject=dataObjectx.createDataObject("FlatFile");

dataObject.set("directoryPath", "C:\\Labfiles60\\eXchange\\FlatFile\\outdir"); dataObject.set("fileName", "publish.txt"); dataObject.set("inputBytes", bytearray);

DataObject do2=(DataObject)service.invoke("create",element);

DataObject outputDataObject = do2.getDataObject("createOutput");//BG

System.out.println(outputDataObject.getDataObject("FlatFile").getString("directoryPath")); System.out.println(outputDataObject.getDataObject("FlatFile").getString("fileName"));

```
}
catch(Throwable t){
t.printStackTrace();
System.out.println("IT FAILED WITH: "+t.toString());
}
```

- \_\_\_\_\_d. Make sure there are no "red x" compile errors. Close the Java file and save the changes.
- 7. Create an SCA export that supports the Publish interface to make this module available to be invoked by other SCA modules.
  - \_\_\_\_a. Right click the Component1 Java component, select Export > SCA Binding.

- ( Eis com/test/flat/FlatFileOutbound	Interface	
🚺 🛃 Componen		
•	💛 Undo Move	
	🕎 Redo	
	Add	▶
	Change Type	•
	Export	<ul> <li>JMS Binding</li> </ul>
	Regenerate Implementation	SCA Binding
	Merge Implementation	Web Service Binding

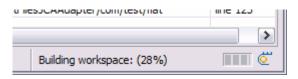
- 8. Change the name and the display name of the newly created **Component1Export** to **PublishFlatFile**.
  - \_\_\_\_a. Single click the Component1Export to select it.
    - \_b. From the Properties pane below the Assembly Diagram, on the Description tab, change the **Display name**: and **Name**: values to **PublishFlatFile.**

Properties 🛛	Problems Servers Console	
Description	🛃 Export: PublishFlatFile (SCA Binding)	
Details	Display name: PublishFlatFile Name: PublishFlatFile	
Binding	Folder: Move	
	Description:	
	Target: Component1	

\_\_\_\_ c. Your module should now look as follows:

■① 📑 com/test/flat/FlatFileOutbo	undInterface
1 E PublishFlatFile	Component1

- 9. Save and close the Assembly Diagram.
  - \_\_\_\_\_a. Close the Assembly Diagram and click Yes at the **Save** Resource popup window to save the changes. In the bottom right corner of the main window, you will see **Building workspace**: and some percentage. Wait for the workspace to complete building.



# Part 3: Test the application using the WebSphere Test Environment (WTE)

In this part you will use the WebSphere Test Environment to test the SCA application outbound processing.

- \_\_\_\_\_1. Start the WebSphere Process Server Test environment.
  - \_\_\_\_a. Right click on the server in the **Servers view** and select **Start**. Wait for the server to be started successfully, as indicated by the Server status of "Started". This could take a few minutes.
- 2. Add the project to the server for the WebSphere Test Environment.
  - \_\_\_\_a. Right click on the server in the server view and select Add and remove projects ...

Properties	Proble	ems 👭 Servers 🛛 Console	
Server		Host	t name
60 WPS Se	erver	New	•
		Open	
		Delete	
		🕸 Debug	
		🜔 Start	
		🔊 Profile	
		Restart	- <b>+</b>
		Stop	
		💦 Disconnect	
		Publish	
	_	Monitoring	<u> </u>
		Add and remove projects	

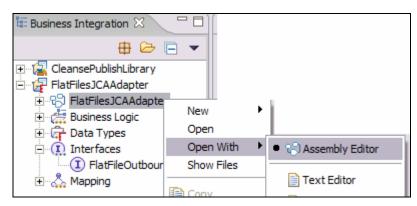
b. In the Add and Remove Projects dialog, select the FlatFilesJCAAdapterApp project from the Available projects panel. Click Add > to add it to the Configured projects panel. Click Finish.

🚰 Add and Remove Projects			
Add and Remove Projects Modify the projects that are config	jured on the server		
Move projects to the right to configu	ire them on the server		
<u>Available projects:</u>		Configured projects:	
			dapterApp
	A <u>d</u> d >		
	< <u>R</u> emove		
	Add All >> << Remove All		
	< <u>B</u> ack <u>N</u> ext >	, <u>E</u> inish	Cancel

\_ c. Wait for the project to be added to the server and the application to start in the server. View the console log from the console view for the application start logs.

WSVR02211: Application started: FlatFilesJCAAdapterApp

- 3. Use Component Test to test the application.
  - \_\_\_\_a. Right click the **FlatFilesJCAAdapter** module and open in the Assembly Editor.



=① = com/test/flat/FlatFile	eOutboundInterface
	ment 1 1.1
	<pre>◇ Redo</pre>
	Add  Change Type Export Regenerate Implementation Merge Implementation Select Implementation Open
	Сору
	Paste
	X Delete Rename Select All
	Wire References to New Wire to Existing Wire (Advanced)
	Test Component

\_\_\_\_b. Right click the Component1 component and select Test Component.

\_\_\_\_ c. Switch from the Events tab to the **Configurations** tab, right click **FlatFileOutboundInterface** under Emulators and click **Remove**. Rather than having Test Component "emulate", we want to use the Java Component implementation code previously entered. There should now be no Emulators, as shown.

🕄 Assembly Diagram: FlatFilesJCAAdapter 🛛 🔲 *Component1_Test	×
Configurations	
Configurations          Test Configuration Default Module Test         Image: Strategy of the	Add Remove Save Load
	-

\_\_\_\_d. Switch back to the **Events** tab, enter whatever values you'd like for the fields of type string, underneath the Clip business object (clipID, GLIN, clip, size, color, brand). For example:

ClipBG String Clip	<null></null>		
-	<null></null>		
Clip			
string	id 1		
string	1234		
string	binder		
string	medium		
string	black		
string	mightyclip 🚽		
ClipItem []	<null></null>		
Data Pool Continue			
	string string string string string		

\_\_\_\_e. Right click retailItems and select Add Element.

brand	string	88
retailItems	Cat Value	<ni< td=""></ni<>
sequence	Set Value	88
	Add Element	
	Remove Element	
	Copy Value	
	Add Value to Pool	
	Use Value from Pool	
	Paste Value	
		_

\_\_\_\_f. Expand the + next to the element to expand twice, enter whatever values you'd like for the string fields, for example:

Initial reguest parameters					
Name		Туре	Value	^	
	itemID	string	item20		
	GTIN	string	7890		
	package	string	pkgA		
	quantity	string	100		
	fullDescrip	string	descriptionhere		
	price	string	20		
	startDate	string	jan 1 2005		
	endDate	string	sept 5 2005	≣	
	contactFir	string	Mary		
	contactLa	string	Smith 🗾	~	
<		1111	>		
Data Pool			Conting	<u>j</u> e	

- \_\_\_g. Select Continue.
- h. Expand the WBI 6.0 Server on the Select Deployment Location panel and select WPS Server v6.0.

Deployment Location		
Select Deployment Location This server instance is currently debugging.		E
Deployment location:		
WBI 6.0 Server WPS Server v6.0		New <u>S</u> erver
Mode: Run	•	
☐ Use this as the default and do not ask again		
	Einish	Cancel

\_\_\_\_i. Select Finish.

**Note:** You may see several exceptions repeated in the Console and SystemOut.log similar to the following. Ignore these exceptions. J2CA0009E: An exception occurred while trying to instantiate the ManagedConnectionFactory class com.ibm.j2ca.flatfile.FlatFileManagedConnectionFactory used by resource FlatFilesJCAAdapter/com/test/flat/FlatFileOutboundInterface\_CF: java.lang.ClassNotFoundException: com.ibm.j2ca.flatfile.FlatFileManagedConnectionFactory

- \_\_\_\_\_4. Verifying Results.
  - \_\_\_\_a. From the **Events** tab, the panels will look as follows:

Events	General Properties	
□-₩ Invoke (Component1:publishClip)	<ul> <li>Detailed Properties</li> </ul>	
E Started	Madda dust as a l	
Invoke (Component1:publishClip) Invoke (Component1> FlatFileOutboundInterface:create)	Module: FlatFilesJCAAdapter	
Response (Component1 < FlatFileOutboundInterface:create)	Component: <u>Component1</u>	
Return (Component 1:publishClip)	Interface: <u>Publish</u>	
Stopped	Operation: publishClip	
	<u>R</u> eturn parameters:	
	Name Type Value	
<		



	Component1_Test - IBM WebSphere Integration Develo Project Run Window Help	oper 📃 💷
		😭 💀 Business Inte
	S Assembly Diagram: FlatFilesJCAAdapter	est 🛛
CleansePublishLibrary	Events  Events  Find the second secon	Image: Second State State Second State State State Second State St
	Events Configurations	verb     VerbType <null>       FlatFile     FlatFile       directoryPath     String     C:\Labfiles60\eXchange       fileName     String     publish.txt       inputBytes     Bytes     [B@758c758c       outputBytes     Bytes     <null>       outputString     String     <null></null></null></null>
	Properties       Problems       Servers       Console       Image: Console <td< td=""></td<>	
	[7/16/05 17:40:30:935 AKDT] 0000004f J2CUt [7/16/05 17:40:34:070 AKDT] 0000004f Syste [7/16/05 17:40:34:070 AKDT] 0000004f Syste	
		<b>&gt;</b>

\_\_\_\_ c. On the file system, located in the specified output directoryPath and fileName specified, there should be a file:

C:\Labfiles60\eXchange\FlatFile\outdir with a file named publish.txt with the following content:

```
<?xml version="1.0" encoding="UTF-8"?>
<scm:Clip xsi:type="scm:Clip"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:scm="http://CleansePublishLibrary/com/clipsandtacks/scm">
```

<clipID>id1</clipID>

<GLN>1234</GLN>

<clip>binder</clip>

<size>medium</size>

<color>black</color>

<brand>mightyclip</brand>

<retailItems>

<itemID>item20</itemID>

<GTIN>7890</GTIN>

<package>pkgA</package>

<quantity>100</quantity>

<fullDescription>descriptionhere</fullDescription>

<price>20</price>

<startDate>jan 1 2005</startDate>

<endDate>sept 5 2005</endDate>

<contactFirstName>Mary</contactFirstName>

<contactLastName>Smith</contactLastName>

</retailItems>

</scm:Clip>

- 5. Clean up test environment.
  - \_\_\_\_a. Close the Component1\_Test panel, click **No** if prompted to save changes. Close the Assembly Diagram panel.
  - \_\_\_\_b. Right-click on WPS Server v6.0 in the Servers view and select Add and remove projects...
  - \_\_\_\_ c. Select FlatFilesJCAAdapter and click < Remove. Click Finish.
  - \_\_\_\_d. Right-click on WPS Server v6.0 and select Stop.

# What you did in this exercise

In this exercise, you imported the WebSphere Adapter for Flat Files and integrated it into an SCA application that creates a file to the file system. You created and implemented a Java component containing the business logic of the application. Component Test was used to input values and test the application.

An SCA export was also created in this exercise, enabling this SCA application to be available to other SCA applications that may want to invoke it.