TIBCO ActiveMatrix® Adapter Service Engine for Tuxedo

Examples

Software Release 6.0
March 2010

Important Information

Contents

Tables

Preface

TIBCO AdtiveMatrix Adapter Servcie Engine for Tuxedo ships with pre-configured examples. This manual describes how to run these examples, and explains the configuration of each.

Topics

Related Documentation

This section lists documentation resources you may find useful.

TIBCO AdtiveMatrix Adapter Servcie Engine for Tuxedo Documentation

The following documents form the TIBCO AdtiveMatrix Adapter Servcie Engine for Tuxedo documentation set:

- *TIBCO AdtiveMatrix Adapter Servcie Engine for Tuxedo Installation* Read this manual to learn how to install TIBCO AdtiveMatrix Adapter Servcie Engine for Tuxedo.
- *TIBCO AdtiveMatrix Adapter Servcie Engine for Tuxedo Configuration and Deployment* Read this manual for instructions on creating, configuring, and deploying adapter projects.
- *TIBCO AdtiveMatrix Adapter Servcie Engine for Tuxedo Examples* Read this manual to work through the examples provided with the adapter.
- *TIBCO AdtiveMatrix Adapter Servcie Engine for Tuxedo Release Notes* Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

Other TIBCO Product Documentation

You may find it useful to read the documentation for the following TIBCO products. Each of the books is available from the `doc` directory in the product's installation area.

- TIBCO Designer™
- TIBCO Administrator™
- TIBCO ActiveMatrix® Administrator
- TIBCO ActiveMatrix BusinessWorks™
- TIBCO ActiveMatrix BusinessWorks™ Service Engine
- TIBCO Rendezvous®
- TIBCO Enterprise Message Service™
- TIBCO Hawk®
- TIBCO Adapter™ SDK
- TIBCO Runtime Agent™

- TIBCO Business Studio™
- TIBCO ActiveMatrix® Service Grid
- TIBCO ActiveMatrix® Service Bus

Third-Party Documentation

You may also find it useful to read the Oracle Tuxedo User documentation.

Typographical Conventions

The following typographical conventions are used in this manual.

Table 1 General Typographical Conventions

Convention	Use
TIBCO_HOME *ENV_HOME* *AMX_HOME*	Many TIBCO products must be installed within the same home directory. This directory is referenced in documentation as *TIBCO_HOME*. The value of *TIBCO_HOME* depends on the operating system. For example, on Windows systems, the default value is `C:\tibco`. Other TIBCO products are installed into an installation environment. Incompatible products and multiple instances of the same product are installed into different installation environments. The directory into which such products are installed is referenced in documentation as *ENV_HOME*. The value of *ENV_HOME* depends on the operating system. For example, on Windows systems the default value is C:\tibco. TIBCO ActiveMatrix installs into a directory inside *ENV_HOME*. This directory is referenced in documentation as *AMX_HOME*. The value of *AMX_HOME* depends on the operating system. For example, on Windows systems the default value is C:\tibco\amx\. TIBCO AdtiveMatrix Adapter Servcie Engine for Tuxedo is installed in a directory inside *AMX_HOME*.
`code font`	Code font identifies commands, code examples, filenames, pathnames, and output displayed in a command window. For example: Use `MyCommand` to start the foo process.
`bold code font`	Bold code font is used in the following ways: • In procedures, to indicate what a user types. For example: Type **`admin`**. • In large code samples, to indicate the parts of the sample that are of particular interest. • In command syntax, to indicate the default parameter for a command. For example, if no parameter is specified, `MyCommand` is enabled: `MyCommand [`**`enable`**` | disable]`

Table 1 General Typographical Conventions (Cont'd)

Convention	Use
italic font	Italic font is used in the following ways: • To indicate a document title. For example: See *TIBCO ActiveMatrix BusinessWorks Concepts*. • To introduce new terms For example: A portal page may contain several portlets. *Portlets* are mini-applications that run in a portal. • To indicate a variable in a command or code syntax that you must replace. For example: `MyCommand` *PathName*
Key combinations	Key name separated by a plus sign indicate keys pressed simultaneously. For example: Ctrl+C. Key names separated by a comma and space indicate keys pressed one after the other. For example: Esc, Ctrl+Q.
	The note icon indicates information that is of special interest or importance, for example, an additional action required only in certain circumstances.
	The tip icon indicates an idea that could be useful, for example, a way to apply the information provided in the current section to achieve a specific result.
	The warning icon indicates the potential for a damaging situation, for example, data loss or corruption if certain steps are taken or not taken.

Table 2 Syntax Typographical Conventions

Convention	Use
`[ ]`	An optional item in a command or code syntax. For example: `MyCommand [optional_parameter] required_parameter`
`|`	A logical `OR` that separates multiple items of which only one may be chosen. For example, you can select only one of the following parameters: `MyCommand para1 | param2 | param3`

Table 2 Syntax Typographical Conventions

Convention	Use

`{ }`	A logical group of items in a command. Other syntax notations may appear within each logical group.
	
	For example, the following command requires two parameters, which can be either the pair `param1` and `param2`, or the pair `param3` and `param4`.
	
	`MyCommand {param1 param2} | {param3 param4}`
	
	In the next example, the command requires two parameters. The first parameter can be either `param1` or `param2` and the second can be either `param3` or `param4`:
	
	`MyCommand {param1 | param2} {param3 | param4}`
	
	In the next example, the command can accept either two or three parameters. The first parameter must be `param1`. You can optionally include `param2` as the second parameter. And the last parameter is either `param3` or `param4`.
	
	`MyCommand param1 [param2] {param3 | param4}`

How to Contact TIBCO Support

For comments or problems with this manual or the software it addresses, please contact TIBCO Support as follows.

- For an overview of TIBCO Support, and information about getting started with TIBCO Support, visit this site:

  http://www.tibco.com/services/support

- If you already have a valid maintenance or support contract, visit this site:

  https://support.tibco.com

  Entry to this site requires a user name and password. If you do not have a user name, you can request one.

Chapter 1 Introduction

This chapter contains an overview of the examples and lists the prerequisites needed to run the examples.

Topics

Overview

The pre-configured examples are located in the *AMX_HOME*\extensions\adtuxedose\samples directory.

- All examples use the FML32 buffer type.
- Some examples obtain the input data from XML files and write the output data into files. And some examples use the client to send data.

Before running the examples, ensure that you change the paths of the input and output XMLs files to the appropriate locations according to your *TIBCO_HOME* installation.

- The following is provided as part of each example:
  - A sample Tuxedo service source code
  - The Tuxedo FML32 header files
  - The ubbconfig files
  - The batch files used to compile and boot the Tuxedo service

  To use the files, you need to decompress the Tuxedo_Adapter_Service_Engine_Samples.zip file to a directory.

Examples List

The following examples are available:

- The RPC Client Binding with JMS example demonstrates how to configure a RPC client in TIBCO Business Studio, it with JMS service, and then run it in TIBCO ActiveMatrix Administrator.
- The RPC Server Binding with TIBCO ActiveMatrix BusinessWorks Service Engine example demonstrates how to configure RPC server in TIBCO Business Studio, it with TIBCO ActiveMatrix BusinessWorks Service Engine service, and then run it in TIBCO ActiveMatrix Administrator.
- The Subscriber Binding with SOAP example demonstrates how to import a TIBCO Desginer project into TIBCO Business Studio. It also demonstrate how to a imported subscriber with SOAP service and run it in TIBCO ActiveMatrix Administrator.
- The EAR TO SA Example demonstrates how to convert an EAR file generated in TIBCO Desginer to a service assembly archive in TIBCO Business Studio, and then run it in TIBCO ActiveMatrix Administrator.

Table 3 shows the communication paradigm, business object and the corresponding adapter service used to develop the examples.

Table 3 Examples Provided with the Adapter

Supported Communication Paradigm	Adapter Service	Business Object
Agent-based Communication	Publisher	PurchaseOrder
Asynchronous Communication	Subscriber	Customer
Synchronous Communication	RPC Server	Employee
Agent-Based Communication	RPC Client	SalesOrder

Prerequisites

Before starting the pre-configured examples, ensure that all the required software has been installed and is operating correctly. For a list of required software, see *TIBCO ActiveMatrix Adapter Service Engine for Tuxedo Installation*.

Using TIBCO ActiveMatrix BusinessWorks Service Engine

When running the examples with TIBCO ActiveMatrix BusinessWorks Service Engine, you need to install TIBCO ActiveMatrix BusinessWorks Service Engine before deploying and running the example. See TIBCO ActiveMatrix BusinessWorks Service Engine documentation for details about how to install TIBCO ActiveMatrix BusinessWorks Service Engine.

Using TIBCO ActiveMatrix Administrator

TIBCO ActiveMatrix Administrator is installed with TIBCO ActiveMatrix Service Grid and is used to deploy and run the examples.

Before deploying and running the examples, ensure that you have set up and configured TIBCO ActiveMatrix Administrator correctly. See TIBCO ActiveMatrix Service Grid documentation for details.

Configuring the Environment for TIBCO ActiveMatrix

Before configuring and running the examples in this manual, if the environment for TIBCO ActiveMatrix has not been configured, follow these steps:

1. Start HSQLDB.
   - Run *AMX_HOME*\amx\hsqldb\bin\amx-db.exe

   or

   - (Windows only) Select **start > All Programs > TIBCO > TIBCO ActiveMatrix** *version_number* **> Start HSQLDB Server**.
2. Start Management Daemon.
   - Run *AMX_HOME*\managementdaemon*version_number*\bin\managementdaemon

   or

   - (Windows only) Select **start > All Programs > TIBCO > TIBCO Management Daemon** *version_number* **> Management Daemon**.
3. Start TIBCO Enterprise Message Service Server
   a. Go to the *EMS_HOME*\bin directory.
   b. Run the following command in the command prompt:

      tibemsd.exe
4. If it is the first time the environment has been configured, create a TIBCO ActiveMatrix Administrator Server using Admin Server Creation Wizard.
5. Start TIBCO ActiveMatrix Administrator.

   Run *AMX_HOME*\amxadministrator*version_number*\bin\amx_admin.bat

If ActiveMatrix Service Grid is installed within TIBCO Runtime Agent, the *AMX_HOME* should be *TIBCO_HOME*.

6. Start the TIBCO ActiveMatrix Administrator Server in a web browser:
   - Run *AMX_HOME*\amxadministrator\data\bootstrap.html

   or

   - (Windows only) Select **start > All Programs > TIBCO > TIBCO ActiveMatrix** *version_number***> Administrator Servers**.

After starting the TIBCO ActiveMatrix Administrator Server, create your admin environment and node with Administrator Server Creation Wizard or in a web browser. The following steps show how to do this in a web browser.

7. Create an Environment in a web browser:
   a. Click **Perspective** and select **Configure Enterprise Assets**.
   b. Click **New** to create a new Environment.
   c. Click on the new environment, and then click **Messaging Bus**.
   d. Click **Add** to create a Messaging Server for each node and save your operation.
   e. Click **Machines** to associate this environment with a machine.
   f. Verify that the containers in the Machines area include the following six types: `SOAP`, `Service Bus Mediation`, `JMS`, `Adapter for Tuxedo`, `Java`, and `.NET`.
8. Create a Node for the Environment in a web browser:
   a. Click **Perspective** and select **Configure an Environment**.
   b. Choose the desired Environment.
   c. Click **New** to create a new node. Enter relational information for the node. The highlighted text boxes are mandatory.
   d. Click **Default Connector** to modify the connector port.
   e. Verify the container for the node.
   f. Click **Install**.

Importing the Projects into TIBCO Business Studio

To view the configuration of the examples, you need to import the ZIP file of the examples to your workspace used by TIBCO Business Studio. To do this, follow these steps:

1. Start TIBCO Business Studio.
2. Select **File > Import...** from the Menu to open the Import dialog.
3. Select **General > Existing Projects into Workspace** in the Select page, and then click **Next**.
4. Select the root directory or the archive file to search in for the existing examples projects, and then click **Next**.
   - Choose between Select root directory and Select archive file and click the **Browse...** button to navigate to the folder where the projects are located or select the archive containing the projects, then click **OK**.
   - In the Projects pane, check the checkbox for the project you want to import as well as the services contained in that project.
   - Check the **Copy projects into workspace** checkbox if you want to copy the projects into the current workspace.
5. Click **Finish** to import the projects.

Preparing Tuxedo

This section describes how to prepare Tuxedo for the pre-configured examples. For more information about preparing Tuxedo, refer to *TIBCO ActiveMatrix BusinessWorks Service Engine Configuration and Deployment*.

The sample tuxedo source code, the ubbconfig files, and the batch files used to compile and boot the Tuxedo service are provided with the pre-configured example. You need to decompress the `Tuxedo_Adapter_Service_Engine_Samples.zip` file to a directory. For example, decompress the `Tuxedo_Adapter_Service_Engine_Samples.zip` file to `D:\demo`. The related files are located in the `Tuxedo Service Codes` sub-directories of each example. You need to set the required variables, boot the Tuxedo services, and ensure that they are running successfully before deploying TIBCO ActiveMatrix Adapter Service Engine for Tuxedo.

To prepare Tuxedo, perform the following tasks:

Task A Edit the make.bat and ubbconfig files

Go to the `Tuxedo Service Codes` sub-directory of the example you want to deploy. For example, if you decompress the `Tuxedo_Adapter_Service_Engine_samples.zip` file to `D:\demo` and want to deploy the RPC Client Binding with JMS example, go to the `D:\demo\NewEclipseUI\Tuxedo Service Codes\SalesOrder` directory. The `make.bat` and `ubbconfig` files are located in the `D:\demo\NewEclipseUI\Tuxedo Service Codes\SalesOrder` directory.

- Editing `make.bat` file

  Modify . For example, set the following values to the variables:

  ```
  set TUXDIR=c:\bea\tuxedo8.1
  set WSNADDR=//192.168.66.44:8100
  set TUX_ADAPTER_HOME=TIBCO_HOME\adapter\adtuxedo\veriosn_number
  set TIBCO_RV_HOME=TIBCO_HOME\tibrv\veriosn_number
  set APPDIR=D:\demo\NewEclipseUI\Tuxedo Service Codes\SalesOrder
  ```

- Editing the `ubbconfig` file

  Modify . For example, set the following values to the variables:

  ```
  *MACHINES
  MYMACHINE LMID=SITE3
  TUXDIR="c:\bea\tuxedo8.1"
  APPDIR="D:\demo\NewEclipseUI\Tuxedo Service Codes\SalesOrder"
  ```

```
TUXCONFIG="D:\demo\NewEclipseUI\Tuxedo Service
Codes\SalesOrder\tuxconfig"
ULOGPFX="D:\demo\NewEclipseUI\Tuxedo Service
Codes\SalesOrder\ulog"
MAXWSCLIENTS=10
WSL SRVGRP=GROUP1 SRVID=3
CLOPT="-A -- -n //192.168.66.44:8100 -m 1 -M 10 -x 1"
```

Task B Boot the Tuxedo Services

Navigate to the APPDIR directory from a command line and run the `make.bat` file as follows. This sets the environment variables, creates the Tuxedo configuration files, creates the executables, and boots the Tuxedo services.

```
D:\demo\NewEclipseUI\Tuxedo Service Codes\SalesOrder make
```

Chapter 2 RPC Client Binding with JMS

Topics

Example Description

This example demonstrates how the RPC client works using the Rendezvous Adapter Agent-based communication paradigm with the `FML32` buffer type.

The the RPC client is d with the JMS service. A TIBCO Designer project is pre-configured to include a TIBCO ActiveMatrix BusinessWorks process used to subscribe a request and returns a response.

The flow of events are as follows.

1. A request based on the Sales Order ID is sent from the Tuxedo client.
2. The Tuxedo client invokes the Tuxedo service.
3. The Tuxedo service sends a request to the adapter through the adapter agent.
4. The agent posts the data to the adapter.
5. The adapter puts the request over JMS transport. TIBCO ActiveMatrix BusinessWorks subscribes to the request and returns a response. After getting the response from TIBCO ActiveMatrix BusinessWorks, the adapter forwards the data to invoke the Tuxedo service through the agent.

Location of the Example

The projects and files used by this example are located in the `NewEclipseUI` folder provided in the `Tuxedo_Adapter_Service_Engine_Samples.zip` file.

Table 4 lists the projects and files used by this example.

Table 4 The Projects and Files Used by the Example

Project or File Name	Description
AMX Projects	
`SalesOrder`	An adapter project containing the configuration information for the adapter instance and RPC client used by the example.
`SalesOrderSOAProject`	An SOA project containing the composite and the service assembly archive (`SalesOrderSOAProject.zip`) for this example. The `SalesOrderSOAProject.zip` file is located in the `Deployment Packages` folder and can be deployed directly in TIBCO ActiveMatrix Administrator.
TIBCO Designer Project	

Table 4 The Projects and Files Used by the Example

Project or File Name	Description
SalesOrderBW	A TIBCO Designer project containing a TIBCO ActiveMatrix BusinessWorks process used to subscribe a request and return a response.
Tuxedo Service Codes	
SalesOrderDetails.h	A Tuxedo header file containing the FML32 definition for the request buffer.
adapteragent.h	A file required to compile the Tuxedo server with the adapter agent. It is a copy of *TIBCO_HOME*\adapter*version_number*\include\adapteragent.h
client.c	A source file for the Tuxedo client and used to invoke the Tuxedo Service ORDERINFO.
inputdata.txt	A file (instead of a database) containing the data that will be used by the server service.
make.bat	A file containing the commands required to get the Tuxedo service running on Windows (Sets the required environment variables, creates executables, loads the ubbconfig file and boots the Tuxedo service)
make.sh	A file containing the commands required to get the Tuxedo service running on UNIX (Sets the required environment variables, creates executables, loads the ubbconfig file, and boots the Tuxedo service). This script should be run in a k-shell.
server.c	A source file for Tuxedo server containing the service ORDERINFO.
ubb	An ubbconfig file

Setting up the Example

To set up the environment for the example, complete the following tasks

- Task A, Prepare Tuxedo
- Task B, Configure the Environment for TIBCO ActiveMatrix
- Task C, Configure a TIBCO Runtime Agent file

To directly deploy the service assembly archive (`SalesOrderSOAProject.zip`) provided with this example, refer to Running the Example. To configure a new adapter instance, create a new SOA project, configure a composite, and create a new service assembly archive for this example, perform Task D through Task I.

- Task D, Create and Configure an Adapter Project
- Task E, Create an SOA Project
- Task F, Generate an Adapter WSDL
- Task G, Create a JMS Shared Resource
- Task H, Configure a Composite
- Task I, Create a Service Assembly Archive

Task A Prepare Tuxedo

Refer to Preparing Tuxedo for detailed information about preparing Tuxedo for the example.

Task B Configure the Environment for TIBCO ActiveMatrix

For detailed information about configuring the Environment for TIBCO ActiveMatrix, refer to Configuring the Environment for TIBCO ActiveMatrix.

Task C Configure a TIBCO Runtime Agent file

The adapter works as a Tuxedo Workstation client. You need to modify the `adtuxedo_wrkstn.tra` file. The file is located in the *TIBCO_HOME*`\adapter\adtuxedo\`*veriosn_number*`\bin` directory. At a minimum, modify :

- `tibco.env.useNewEclipseUI`— set it to `ON`.
- `tibco.env.TUXConnNumber`— the number of connections with the Tuxedo application. The default value is 10.

- `tibco.env.RVAgentThreadNumber`— the number of threads to be used by the Rendezvous agent-based publisher. The default value is 4.
- `tibco.env.WSNADDR`—The IP address of the WSL (Workstation Listener) that the client will contact, along with a free port number. This value must be reflected in the `ubbconfig` file.

Refer to *TIBCO ActiveMatrix BusinessWorks Service Engine Configuration and Deployment* for detailed information about TIBCO Runtime Agent file.

Task D Create and Configure an Adapter Project

1. Create an adapter project.
   a. Start TIBCO Business Studio, select **New >Adapter Resources...** from the File menu to open the Create New Adapters Resource dialog.
   b. Select **Adapter Project** and click **Next**.
   c. Type the project name in the New Adapter Project dialog. For example, type **`SalesOrder`**.
   d. Click **Finish**. An adapter project is created.
2. Create and Configure a Tuxedo Connection.
   a. Click the `SalesOrder` project in the Project Explorer Panel, and then select **File > New > Adapter Resources...** from the Menu to open the Create New Adapters Resource dialog.
   b. Expand the **Shared Resources** folder and select **Tuxedo Connection**.
   c. Click **Next** to open the Tuxedo Connection Shared Resource dialog, and then select **`SalesOrder`** as the parent folder.
   d. Click **Finish.**

3. Import Tuxedo BOs

   a. Select the `SalesOrder` project in the Project Explorer panel, and then select **File > Import** from the Menu.

   b. Expand the **TIBCO ActiveMatrix Adapters** folder and select **Metadata > Tuxedo Objects**.

   c. Click **Next** to open the Tuxedo Business Object Configuration dialog.

   d. In the Tuxedo Business Object Configuration dialog, enter **`SalesOrderDetails`** in the Name field and browse to the `SalesOrder` project in the Download Location field.

   e. Specify the header file to be used in the Tuxedo Header File Path field.

      For this example, navigate to the `Tuxedo Service Codes\SalesOrder` sub-directory of this example and select the `SalesOrderDetails.h` header file.

   f. Click **Next**.

   g. In the Tuxedo Business Object Browser dialog, select **Tuxedo > TuxedoGroup** in the Applications panel and then click the **Select All** button to select all of the fields.

   h. Click **Finish**. The Tuxedo BO to be used by this example is imported.

4. Create and Configure a Tuxedo Adapter Configuration

   a. Click the `SalesOrder` project in the Project Explorer Panel, and then select **File > New > Adapter Resources...** from the Menu to open the Create New Adapters Resource dialog.

   b. Expand the **Configurations** folder and select **Tuxedo Adapter**. Click **Next** to open the Tuxedo Adapter Configuration dialog.

   c. Select `SalesOrder` as the parent folder.

   d. Click **Finish**. The configuration file `TuxedoAdapterConfiguration.adtuxedomodel` is under the `SalesOrder` project in the Project Explorer Panel.

   e. Double-click `TuxedoAdapterConfiguration.adtuxedomodel` to open the configuration tabs on the right Editor Panel.

   f. In the Run-time Connection area, select **Browse** from the Connection Configuration drop-down list to specify the Tuxedo connection created in step 2.

5. Create and Configure a RPC client
   a. Click the **Adapter Services** tab.
   b. Click the **Add Request Response Invocation Service** button in the All Adapter Services pane. The parameter configuration pane appears in the tab.
   c. In the Configuration area, specify the service name in the Name field. Make sure that the service name is same as the `serviceName` parameter used by the `callClient()` method.
   d. In the Schema area, select **Browse** from the Request Class Reference From BO and Reply Class Reference From BO drop-down lists to specify the Tuxedo BO imported in step 3 and then click **Apply**.
   e. In the RPC Client Option area, select **FML32** in the Buffer Type drop-down list and **Rendezvous** in the Agent Transport Type drop-down list.

Task E Create an SOA Project

1. Select **File > New > Project...** from the Menu to open the New Project dialog.
2. Expand **TIBCO ActiveMatrix > ActiveMatrix SOA Project**, and then click **Next**.
3. Type **`SalesOrderSOAProject`** in the Project name field, and then click **Next**.
4. Click **Finish**.

Task F Generate an Adapter WSDL

1. Right-click `TuxedoAdapterConfiguration.adtuxedomodel` under the `SalesOrder` project in the Project Explorer Panel, and then select **Generate Adapter WSDL** from the pop-up menu. The Target Project dialog is opened.
2. Select `SalesOrderSOAProject` as the matching resource, and then click **OK**.

Task G Create a JMS Shared Resource

1. Select the `SalesOrderSOAProject` project in the Project Explorer Panel, and then select **File > New > TIBCO Shared Resources...** from the Menu. The TIBCO Resource Wizard dialog is opened.
2. Select **JMS** and click **Finish**.

Task H Configure a Composite

1. Launch the Composite Editor.
   a. Expand **`SalesOrderSOAProject`** > **Composites** in the Project Explorer panel.
   b. Double-click **`SalesOrderSOAProject.composite`** under the Composites folder to launch the Composite Editor on the right panel.
2. Create and configure a Tuxedo adapter component.
   a. Drag the **TuxedoAdapter** component from the Palette to the Components column and enter **`Adapter`** as the component name.
   b. Click the **`Adapter`** component, the configuration parameters are displayed under the Properties Views panel.

      In the service tab, click the **Add** button to add the RPC client endpoint to the component.
3. Add the JMS Shared Resource to Composite.
   a. Click the canvas in the Composite Editor to open the Composite Properties Views panel under the Composite Editor.
   b. Click the **Shared Resource Profiles** tab, add the JMS shared resource by clicking  .
   c. Select **JMS** in the Type drop-down list.
   d. Click the Target column and click  . The Select Shared Resources dialog is opened.
   e. Select the JMS shared resources created in Task G.
4. Add and Configure a JMS Service
   a. Drag a JMS service from the Palette to the Services column and enter **`Jms`** as the service name.
   b. Click the `Jms` service, the configuration parameters are displayed under the Properties Views panel.
   c. Click the **Target** tab and click the radio button next to the target component service to draw a between the `Jms` service and the `Adapter` component.
   d. Click the **Binding** tab to add the shared resource profile to the Endpoint pane.
5. Save the project.

Task I Create a Service Assembly Archive

1. Right-click `SalesOrderSOAProject.composite` under the Composites folder and select **Service Assembly** from the pop-up menu.
2. Save the composite if prompted. `SalesOrderSOAProject.saf` is created in the `Deployment` Packages folder.
3. Right-click `SalesOrderSOAProject.saf`. and select **Build Archive** from the pop-up menu to create the `SalesOrderSOAProject.zip` file.

The `SalesOrderSOAProject.zip` file is ready to be deployed in TIBCO ActiveMatrix Administrator.

Running the Example

To run the example, you need to complete the following tasks:

- Task A, Deploy an SA File, page 20
- Task B, Start the Node and Installing the Shared Resources, page 21
- Task C, Deploy and Start the SA Project, page 21
- Task D, Run the Process Included in the TIBCO Desginer Project, page 21

Task A Deploy an SA File

Before proceeding, ensure that the HSQLDB, Management Daemon, TIBCO ActiveMatrix Administrator, TIBCO ActiveMatrix Administrator Server, TIBCO Enterprise Message Service Server, and Tuxedo service are running.

To deploy the SA project (the ZIP file) in TIBCO ActiveMatrix Administrator:

1. Select the **Deploy to an Environment** perspective.
2. Choose the Environment, and then click the **Upload Service Assembly** button.
3. Enter a name for the SA in the Name field.
4. Click the **Browse** button to select the archived ZIP file (`SalesOrderSOAProject.zip`).
5. Check the **Import Shared Resource Definitions** checkbox to create the shared resources definitions.
6. Click **OK**.
7. In the Service Assemblies panel, select the service assembly.
8. Click the **Service Units** button.
9. For each service unit in the service assembly, follow these steps:
    a. Select a service unit in the Service Units table.
    b. In the Node Mapping tab, select the nodes which you want to deploy the service unit to from the Available Nodes list, and then click the right arrow button. The nodes will appear in the Mapped Nodes list.
    c. Click **Save**.

Task B Start the Node and Installing the Shared Resources

If an ActiveMatrix Service (JMS, SOAP, or AMX) is included in the SA project, you need to install the corresponding Shared Resource for the appropriate node.

1. In the web browser, select **Configure an Environment** in the Perspective drop-down list, and then select the appropriate Environment.
2. In the Nodes table, select the appropriate node, then click **Start**.
3. In the Node Detail panel, select **Shared Resources**.
4. Enable the corresponding shared resource and install it.

Task C Deploy and Start the SA Project

1. Select the SA Project in TIBCO ActiveMatrix Administrator, and then click **Deploy**.
2. Ensure that the status of the SA project is set to `deployed`. Select the SA project, and then click **Start**.

Task D Run the Process Included in the TIBCO Desginer Project

1. Open the `SaleOrderBW` project in TIBCO Desginer.
2. Select the `rpcc` process in the project panel.
3. Click the **Tester** tab in the project panel.
4. Click the **Start testing viewed process** button to open the Select Process to Load dialog.
5. Ensure that the `rpcc` process is selected.
6. Click **Load Selected**.

Expected Results

In a command window, start the Tuxedo Client and specify the `Order ID`. For example, specify the `Order ID` as follows:

your_directory`\NewEclipseUI\Tuxedo Service Codes\SalesOrder>client 1000`

The output is displayed in the client window. As an example, the following data is returned based on the `Order ID` 1000:

```
ORDER NO=1000
PRODUCT ID=1234
DESCRIPTION=BOOKS
ORDER STATUS=ACTIVE

UNIT OF MEASURE=NO
ORDER DATE=29-MAR-2003
PRODUCT QTY ORDERED=12.000000
```

Chapter 3

RPC Server Binding with TIBCO ActiveMatrix BusinessWorks Service Engine

Topics

Example Description

This example demonstrates how the RPC server works using the Synchronous communication paradigm with the FML32 buffer type. The the RPC server is d with TIBCO ActiveMatrix BusinessWorks Service Engine.

The flow of events are as follows.

1. Change the value of the Department ID in the Get_input.xml file.
2. TIBCO ActiveMatrix BusinessWorks Service Engine reads the input data and sends the request. The File Poller activity in the process reads the data and forwards it to the Invoke Partner activity.
3. The Invoke Partner activity then invokes the EMPINFO Tuxedo Service.
4. The Tuxedo service queries the inputdata.txt file based on the department ID that is received.
5. The Tuxedo service then returns the records that match the query.
6. The process writes the data into the Get_output.xml file.

Location of the Example

The projects and files used by this example are located in the BWSE folder provided in the Tuxedo_Adapter_Service_Engine_Samples.zip file.

Table 5 lists the projects and files used by this example.

Table 5 The Projects and Files Used by the Example

Project or File Name	Description
AMX Projects	
EmployeeRpc	A adapter project containing the configuration information for the adapter instance and RPC server used by the example.
EmployeeRpcSOAProject	An SOA project containing the composite and the service assembly archive (EmployeeRpcSOAProject.zip) for this example. The EmployeeRpcSOAProject.zip file is located in the Deployment Packages folder and can be deployed directly in TIBCO ActiveMatrix Administrator.
EmployeeRpcDesignerProject	A TIBCO Desginer project created in TIBCO Business Studio. It contains the TIBCO ActiveMatrix BusinessWorks Service Engine process used by this example.

Table 5 The Projects and Files Used by the Example

Project or File Name	Description
Tuxedo Service Codes	
`EmployeeDetailsInput.h`	A Tuxedo header file containing the FML32 definition for the request buffer.
`EmployeeDetailsOutput.h`	A Tuxedo header file containing the FML32 definition for the response buffer.
`Get_input.xml`	A file containing request data in XML format.
`inputdata.txt`	A file (instead of a database) containing the data that will be used by the server service.
`make.bat`	A file containing the commands required to get the Tuxedo service running on Windows. It sets the required environment variables, creates executables, loads the `ubbconfig` file, and boots the Tuxedo service.
`make.sh`	A file containing the commands required to get the Tuxedo service running on UNIX. It sets the required environment variables, creates executables, loads the `ubbconfig` file, and boots the Tuxedo service. This script should be run in a k-shell.
`server.c`	A source file for Tuxedo Server containing the service `EMPINFO`
`ubbrpc`	An `ubbconfig` file

Setting up the Example

To set up the environment for the example, complete the following tasks

- Task A, Prepare Tuxedo
- Task B, Configure the Environment for TIBCO ActiveMatrix
- Task C, Configure a TIBCO Runtime Agent file

To directly deploy the service assembly archive (`EmployeeRpcSOAProject.zip`) provided with this example, refer to Running the Example. To configure a new adapter instance, create a new SOA project, configure a composite, and create a new service assembly archive for this example, perform Task D through Task K.

- Task D, Create and Configure an Adapter Project
- Task E, Create an SOA Project
- Task F, Create a TIBCO Desginer Project
- Task G, Generate the Adapter WSDL
- Task H, Import Resources to the TIBCO Designer Project
- Task I, Configure the SOA Project
- Task J, Configure the BusinessWorks Process in the Designer Project
- Task K, Create a Service Assembly Archive

Task A Prepare Tuxedo

Refer to Preparing Tuxedo for detailed information about preparing Tuxedo for the example.

Task B Configure the Environment for TIBCO ActiveMatrix

For detailed information about configuring the Environment for TIBCO ActiveMatrix, refer to Configuring the Environment for TIBCO ActiveMatrix.

Task C Configure a TIBCO Runtime Agent file

The adapter works as a Tuxedo Workstation client. You need to modify the `adtuxedo_wrkstn.tra` file. The file is located in the *TIBCO_HOME*`\adapter\adtuxedo\`*veriosn_number*`\bin` directory. At a minimum, modify :

- `tibco.env.useNewEclipseUI`— set it to `ON`.

- `tibco.env.TUXConnNumber`— the number of connections with the Tuxedo application. The default value is 10.
- `tibco.env.WSNADDR`—The IP address of the WSL (Workstation Listener) that the client will contact, along with a free port number. This value must be reflected in the `ubbconfig` file.

Refer to *TIBCO ActiveMatrix BusinessWorks Service Engine Configuration and Deployment* for detailed information about TIBCO Runtime Agent file.

Task D Create and Configure an Adapter Project

1. Create an adapter project.
   a. Start TIBCO Business Studio, select **New >Adapter Resources...** from the File menu to open the Create New Adapters Resource dialog.
   b. Select **Adapter Project** and click **Next**.
   c. Type the project name in the New Adapter Project dialog. For example, type **`EmployeeRpc`**.
   d. Click **Finish**. An adapter project is created.
2. Create and Configure a Tuxedo Connection.
   a. Click the **`EmployeeRpc`** project in the Project Explorer Panel, and then select **File > New > Adapter Resources...** from the Menu to open the Create New Adapters Resource dialog.
   b. Expand the **Shared Resources** folder and select **Tuxedo Connection**.
   c. Click **Next** to open the Tuxedo Connection Shared Resource dialog, and then select **`EmployeeRpc`** as the parent folder.
   d. Click **Finish.**

3. Import Tuxedo BOs

   a. Select the **`EmployeeRpc`** project in the Project Explorer panel, and then select **File > Import** from the Menu.

   b. Expand the **TIBCO ActiveMatrix Adapters** folder and select **Metadata > Tuxedo Objects**.

   c. Click **Next** to open the Tuxedo Business Object Configuration dialog.

   d. In the Tuxedo Business Object Configuration dialog, enter **`EmployeeInputDetails`** in the Name field and browse to the `EmployeeRpc` project in the Download Location field.

   e. Specify the header file to be used as the request class reference in the Tuxedo Header File Path field.

      For this example, navigate to the `Tuxedo Service Codes\Employee` sub-directory of this example and select the `EmployeeDetailsInput.h` header file.

   f. Click **Next**.

   g. In the Tuxedo Business Object Browser dialog, select **Tuxedo > TuxedoGroup** in the Applications panel and then click the **Select All** button to select all of the fields.

   h. Click **Finish**. The Tuxedo BO to be used by this example is imported.

   i. Repeat step a through step h to import the `EmployeeDetailsOutput.h` file as the reply class reference and name the BO `EmployeeOutputDetails.bo`.

4. Create and Configure a Tuxedo Adapter Configuration
   a. Click the **`EmployeeRpc`** project in the Project Explorer Panel, and then select **File > New > Adapter Resources...** from the Menu to open the Create New Adapters Resource dialog.
   b. Expand the **Configurations** folder and select **Tuxedo Adapter**. Click **Next** to open the Tuxedo Adapter Configuration dialog.
   c. Select **`EmployeeRpc`** as the parent folder.
   d. Click **Finish**. The configuration file `TuxedoAdapterConfiguration.adtuxedomodel` is under the `EmployeeRpc` project in the Project Explorer Panel.
   e. Double-click `TuxedoAdapterConfiguration.adtuxedomodel` to open the configuration tabs on the right Editor Panel.
   f. In the Run-time Connection area, select **Browse** from the Connection Configuration drop-down list to specify the Tuxedo connection created in step 2.
5. Create and Configure a RPC server
   a. Click the **Adapter Services** tab.
   b. Click the **Add Request Response Service** button in the All Adapter Services pane. The parameter configuration pane appears in the tab.
   c. In the Schema area, select **Browse** from the Request Class Reference From BO drop-down list to specify `EmployeeInputDetails.bo`.

      Select **Browse** from the Reply Class Reference From BO drop-down list to specify `EmployeeOutputDetails.bo`. Click **Apply**.
   d. In the RPC Server Option area, select **FML32** in the Buffer Type drop-down list.

Task E Create an SOA Project

1. Select **File > New > Project...** from the Menu to open the New Project dialog.
2. Expand **TIBCO ActiveMatrix > ActiveMatrix SOA Project**, and then click **Next**.
3. Type **`EmployRpcSOAProject`** in the Project name field, and then click **Next**.
4. Click **Finish**.

Task F Create a TIBCO Desginer Project

1. Select **File > New > Project** from the Menu to open the New Project dialog.

2. Select **TIBCO Designer > TIBCO Designer Project** in the Select a wizard page, and then click **Next**.
3. Type **`EmployRpcDesignerProject`** in the Project name field, and select the **Create new project in the workspace** radio button.
4. Click **Finish**.

Task G Generate the Adapter WSDL

1. Right-click `TuxedoAdapterConfiguration.adtuxedomodel` under the `EmployeeRpc` project in the Project Explorer Panel, and then select **Generate Adapter WSDL** from the pop-up menu. The Target Project dialog is opened.
2. Select **`EmployRpcSOAProject`** as the matching resource, and then click **OK**.

Task H Import Resources to the TIBCO Designer Project

1. In the Project Explorer Panel, select **`EmployRpcDesignerProject`**.
2. Select **File > Import...** from the Menu to open the Import dialog.
3. Expand **General folder** > **File System**, and then click **Next**.
4. Click **Browse** to select the `EmployRpcSOAProject` project in the From directory field, and check the **TuxedoAdapterConfiguration_genresources** and **schema** checkboxes in the left pane.
5. Click **Finish**.

Task I Configure the SOA Project

1. In the Project Explorer Panel, select **`EmployRpcSOAProject`**, expand the **Composites** folder, and then double-click **`EmployRpcSOAProject.composite`** to launch the Composite Editor on the right.
2. Create and configure a TuxedoAdapter component.
   a. Drag the **TuxedoAdapter** component from the Palette to the Components column and enter **`RpcAdapter`** as the component name.
   b. Click **`RpcAdapter`**, the configuration parameters are displayed under the Properties Views panel.

      In the service tab, click **Add** to add the RPC server endpoint to the component.

3. Create and configure a BusinessWorks component.
   a. Drag the **BusinessWorks** component from the Palette to the Components column and enter **`BW`** as the component name.
   b. In the Implementation tab, click **Browse** to select the **`EmployRpcDesignerProject`** project in the BW Project field.
   c. Click the **Reference** tab in the Properties view.

      Click **Add** to add the service reference that you have imported in Task H. The BusinessWorks process, `TuxedoAdapterConfiguration_RequestResponseService.process`, is generated in `EmployRpcDesignerProject`.

      Click the radio button next to `RpcAdapter->RequestResponseService` to draw a between `RpcAdapter` and `BW`.

Task J Configure the BusinessWorks Process in the Designer Project

1. Double-click **`TuxedoAdapterConfiguration_RequestResponseService.process`** under the `EmployRpcDesignerProject` project to open the Editor View on the right pane.
2. Create and configure a File Poller activity.
   a. Drag and drop a File Poller activity from the Palettes panel to the Design panel.
   b. In the Configuration tab, specify the `Get_Imput.xml` file to be used in the File Name field.
3. Create and configure a Invoke Partner activity.
   a. Drag and drop a Invoke Partner activity from the Palettes panel to the Design panel.
   b. In the Configuration tab, select **`Partner_bwse_1`** from the Partner drop-down list.
   c. In the Input tab, expand **`operation_request > request > _carnet_request_carnet_RequestResponseServiceRPCClass_carnet _operation>input`** tree. Specify the department ID in the DEPTID field. Enter **`"EMPINFO"`** in the ServiceName field.

4. Create and configure a Write File activity.
   a. Drag and drop a Write File activity from the Palettes panel to the Design panel.
   b. In the Input tab, map the Invoke Partner output (a string called `operation_response > response > _carnet_reply_carnet_RequestResponseServiceRPCClass_carnet_operation > EMPNAME_167772262`) to the incoming Write File schemas (a string named `WriteActivityInputTextClass > textContent`)
5. Connect the File Poller, Invoke Partner, and Write File activities.

Task K Create a Service Assembly Archive

1. Right-click **`EmployeeRpcSOAProject.composite`** and select **Service Assembly** from the pop-up menu.
2. Save the composite if prompted. The `EmployRpcSOAProject.saf` file is created in the `Deployment Packages` folder.
3. Right-click **`EmployeeRpcSOAProject.saf`**. and select **Build Archive** from the pop-up menu. The `EmployeeRpcSOAProject.zip` file is created.

The `EmployeeRpcSOAProject.zip` is ready to be deployed in TIBCO ActiveMatrix Administrator.

Running the Example

To run the example, you need to complete the following tasks:

- Task A, Deploy an SA File
- Task B, Start the Node
- Task C, Deploy and Start the SA Project
- Task D, Trigger the RPC Server, page 34

Task A Deploy an SA File

Before proceeding, ensure that the HSQLDB, Management Daemon, TIBCO ActiveMatrix Administrator, TIBCO ActiveMatrix Administrator Server, TIBCO Enterprise Message Service Server, and Tuxedo service are running.

To deploy the SA project (the ZIP file) in TIBCO ActiveMatrix Administrator:

1. Select the **Deploy to an Environment** perspective.
2. Choose the Environment, and then click the **Upload Service Assembly** button.
3. Enter a name for the SA in the Name field.
4. Click the **Browse** button to select the archived ZIP file (`EmployeeRpcSOAProject.zip`).
5. Check the **Import Shared Resource Definitions** checkbox to create the shared resources definitions.
6. Click **OK**.
7. In the Service Assemblies panel, select the service assembly.
8. Click the **Service Units** button.
9. For each service unit in the service assembly, follow these steps:
   a. Select a service unit in the Service Units table.
   b. In the Node Mapping tab, select the nodes which you want to deploy the service unit to from the Available Nodes list, and then click the right arrow button. The nodes will appear in the Mapped Nodes list.
   c. Click **Save**.

Task B Start the Node

1. In the web browser, select **Configure an Environment** in the Perspective drop-down list, and then select the appropriate Environment.
2. In the Nodes table, select the appropriate node, then click **Start**.

Task C Deploy and Start the SA Project

1. Select the SA Project in TIBCO ActiveMatrix Administrator, and then click **Deploy**.
2. Ensure that the status of the SA project is set to `deployed`. Select the SA project, and then click **Start**.

Task D Trigger the RPC Server

Open the `Get_input.xml` file and change the value of the department ID. For example, change the department ID as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<Employees xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Employee>
    <DEPTID>123</DEPTID>
  </Employee>
</Employees>
```

Expected Results

After changing the department ID in the `Get_input.xml` file, the output is written in the `Get_output.xml` file. The `Get_output.xml` file is located in the *TIBCO_HOME*`\adapter\adtuxedo\`*version_number*`\examples\Synchronous\Static\Employee` directory.

Chapter 4 Subscriber Binding with SOAP

Topics

Example Description

This example demonstrates how to import a TIBCO Desginer project into TIBCO Business Studio. It also demonstrate how to a imported subscriber with SOAP service. The subscriber works using the asynchronous communication paradigm with FML32 buffers.

The CustumerSubBW TIBCO Desginer project is pre-configured and includes a TIBCO ActiveMatrix BusinessWorks process to send the data on the subject that the subscriber is subscribing to.

The flow of events is as follows:

1. The values for CUSTID, NAME, ADDRESS, CITY, STATE, ZIP, or PHONE in the Get_input.xml file are changed.
2. The TIBCO ActiveMatrix BusinessWorks process is triggered. The File Poller activity reads the data and forwards it to the SOAP Request Reply activity.
3. The SOAP Request Reply activity performs the request and sends the data to the adapter.
4. The subscriber, which is listening on the same subject as that of the invoking client (TIBCO ActiveMatrix BusinessWorks), subscribes to the subject and gets the data.
5. The subscriber invokes the NEWCUST Tuxedo service.
6. The NEWCUST Tuxedo services writes the data to the CustomerDetails file based on the received data.

Location of the Example

The projects and files used by this example are located in the Import folder provided in the Tuxedo_Adapter_Service_Engine_Samples.zip file.

Table 6 lists the projects and files used by this example.

Table 6 The Projects and Files Used by the Example

Project or File Name	Description
AMX Projects	
CustumerSub	An adapter project that is imported from the CustumerSub TIBCO Desginer project. The adapter instance and subscriber in this project are configured in TIBCO Desginer and imported into TIBCO Business Studio.

Table 6 The Projects and Files Used by the Example

Project or File Name	Description
CustumerSubSOAProject	An SOA project containing the composite and the service assembly archive (CustumerSubSOAProject.zip) for this example. The CustumerSubSOAProject.zip file is located in the Deployment Packages folder and can be deployed directly in TIBCO ActiveMatrix Administrator.
TIBCO Desginer Projects	
CustumerSub	A TIBCO Desginer project containing the configuration information for the adapter instance and subscriber used by the example.
CustumerSubBW	A TIBCO Designer project containing a TIBCO ActiveMatrix BusinessWorks process used to subscribe a request.
Tuxedo Service Codes	
CustDetails.h	A Tuxedo header file containing the FML32 definition.
Get_input.xml	A file containing the input data in the XML format.
Services.h	A header file required by the sample Tuxedo server program.
make.bat	A file containing the commands required to get the Tuxedo service running on Windows. It sets the required environment variables, creates executables, loads the ubbconfig file, and boots the Tuxedo service.
make.sh	A file containing the commands required to get the Tuxedo service running on UNIX. It sets the required environment variables, creates executables, loads the ubbconfig file, and boots the Tuxedo service. This script should be run in a k-shell.
server.c	A source file for Tuxedo Server contains the service NEWCUSTID
ubbAsyn	An ubbconfig file
util.c	A source file used by the Tuxedo service

Setting up the Example

To set up the environment for the example, complete the following tasks

- Task A, Prepare Tuxedo
- Task B, Configure the Environment for TIBCO ActiveMatrix
- Task C, Configure a TIBCO Runtime Agent file

To directly deploy the service assembly archive (`CustomerSubSOAProject.zip`) provided with this example, refer to Running the Example. To import the pre-configured TIBCO Desginer project, create a new SOA project, configure a composite, and create a new service assembly archive for this example, perform Task D through Task I.

- Task D, Import the Pre-configured TIBCO Desginer Project into TIBCO Business Studio
- Task E, Create an SOA Project
- Task F, Generate an Adapter WSDL
- Task G, Create a HTTP Server Shared Resource
- Task H, Configure a Composite
- Task I, Create a Service Assembly Archive

Task A Prepare Tuxedo

Refer to Preparing Tuxedo for detailed information about preparing Tuxedo for the example.

Task B Configure the Environment for TIBCO ActiveMatrix

For detailed information about configuring the Environment for TIBCO ActiveMatrix, refer to Configuring the Environment for TIBCO ActiveMatrix.

Task C Configure a TIBCO Runtime Agent file

The adapter works as a Tuxedo Workstation client. You need to modify the `adtuxedo_wrkstn.tra` file. The file is located in the *TIBCO_HOME*`\adapter\adtuxedo\`*veriosn_number*`\bin` directory. At a minimum, modify :

- `tibco.env.useNewEclipseUI`— set it to `ON`.

- `tibco.env.TUXConnNumber`— the number of connections with the Tuxedo application. The default value is 10.
- `tibco.env.WSNADDR`—The IP address of the WSL (Workstation Listener) that the client will contact, along with a free port number. This value must be reflected in the `ubbconfig` file.

Refer to *TIBCO ActiveMatrix BusinessWorks Service Engine Configuration and Deployment* for detailed information about TIBCO Runtime Agent file.

Task D Import the Pre-configured TIBCO Desginer Project into TIBCO Business Studio

1. Start TIBCO Business Studio.
2. Select **File > Import...** from the Menu to open the Import window.
3. Expand the **TIBCO ActiveMatrix Adapters** folder and select **Import TIBCO Designer Projects** in the Select page.
4. Select the directory where the `CustumerSub` TIBCO Desginer projects is located.
5. In the Projects pane, check the checkbox for the project to be imported as well as the adapter instance and subscriber contained in the project.
6. Click **Finish**.

Task E Create an SOA Project

1. Select **File > New > Project...** from the Menu to open the New Project dialog.
2. Expand **TIBCO ActiveMatrix > ActiveMatrix SOA Project**, and then click **Next**.
3. Type **`CustomerSubSOAProject`** in the Project name field, and then click **Next**.
4. Click **Finish**.

Task F Generate an Adapter WSDL

1. Right-click `TuxedoAdapterConfiguration.adtuxedomodel` under the `CustomerSub` project in the Project Explorer Panel, and then select **Generate Adapter WSDL** from the pop-up menu. The Target Project dialog is opened.
2. Select **`CustomerSubSOAProject`** as the matching resource, and then click **OK**.

Task G Create a HTTP Server Shared Resource

1. Select **`CustomerSubSOAProject`** in the Project Explorer Panel, and then select **File > New > TIBCO Shared Resources...** from the Menu to open the TIBCO Resource Wizard dialog.
2. Select **HTTP Server** and click **Finish**.

Make sure that the port number used by the HTTP server is unique.

Task H Configure a Composite

1. Launch the Composite Editor.
   a. Expand **`CustomerSubSOAProject`** > **Composites** in the Project Explorer panel.
   b. Double-click **`CustomerSubSOAProject.composite`** under the Composites folder to launch the Composite Editor on the right panel.
2. Create and configure a Tuxedo adapter component.
   a. Drag the **TuxedoAdapter** component from the Palette to the Components column and enter **`TuxedoAdapter`** as the component name.
   b. Click the **`TuxedoAdapter`** component, the configuration parameters are displayed under the Properties Views panel.

      In the service tab, click the **Add** button to add the subscriber endpoint to the component.
3. Add the HTTP Server Shared Resource to Composite.
   a. Click the canvas in the Composite Editor to open the Composite Properties Views panel under the Composite Editor.
   b. Click the **Shared Resource Profiles** tab, add the HTTP Server shared resource by clicking  .
   c. Select **HTTP** in the Type drop-down list.
   d. Click the Target column and click  to open the Select Shared Resources dialog.
   e. Select the HTTP Server shared resources created in Task G.

4. Add and Configure a SOAP Service

   a. Drag a SOAP service from the Palette to the Services column and enter **`Soap`** as the service name.

   b. Click the **`Soap`** service, the configuration parameters are displayed under the Properties Views panel.

   c. In the Target tab, click the radio button next to the target component service to draw a between the `Soap` service and the `TuxdoAdapter` component.

   d. In the Binding tab, click the **Generate WSDL** button to generate a WSDL file named `TuxedoAdapterConfiguration_SubscriptionService_`*`Number`*`.wsdl`.

      This WSDL file is used to configure the TIBCO ActiveMatrix BusinessWorks process in the `CustomerSubBW` TIBCO Desginer project.

5. Save the project.

Task I Create a Service Assembly Archive

1. Right-click **`CustomerSubSOAProject.composite`** under the Composites folder and select **Service Assembly** from the pop-up menu.
2. Save the composite if prompted. The `CustomerSubSOAProject.saf` file is created in the `Deployment` Packages folder.
3. Right-click **`CustomerSubSOAProject.saf`** and select **Build Archive** from the pop-up menu. `CustomerSubSOAProject.zip` is created.

The `CustomerSubSOAProject.zip` is ready to be deployed in TIBCO ActiveMatrix Administrator.

Running the Example

To run the example, you need to complete the following tasks:

- Task A, Deploy an SA File
- Task B, Start the Node and Installing the Shared Resources
- Task C, Deploy and Start the SA Project
- Task D, Run the Process Included in the TIBCO Desginer Project

Task A Deploy an SA File

Before proceeding, ensure that the HSQLDB, Management Daemon, TIBCO ActiveMatrix Administrator, TIBCO ActiveMatrix Administrator Server, TIBCO Enterprise Message Service Server, and Tuxedo service are running.

To deploy the SA project (the ZIP file) in TIBCO ActiveMatrix Administrator:

1. Select the **Deploy to an Environment** perspective.
2. Choose the Environment, and then click the **Upload Service Assembly** button.
3. Enter a name for the SA in the Name field.
4. Click the **Browse** button to select the archived ZIP file (`CustomerSubSOAProject.zip`).
5. Check the **Import Shared Resource Definitions** checkbox to create the shared resources definitions.
6. Click **OK**.
7. In the Service Assemblies panel, select the service assembly.
8. Click the **Service Units** button.
9. For each service unit in the service assembly, follow these steps:
    a. Select a service unit in the Service Units table.
    b. In the Node Mapping tab, select the nodes which you want to deploy the service unit to from the Available Nodes list, and then click the right arrow button. The nodes will appear in the Mapped Nodes list.
    c. Click **Save**.

Task B Start the Node and Installing the Shared Resources

If an ActiveMatrix Service (JMS, SOAP, or AMX) is included in the SA project, you need to install the corresponding Shared Resource for the appropriate node.

1. In the web browser, select **Configure an Environment** in the Perspective drop-down list, and then select the appropriate Environment.
2. In the Nodes table, select the appropriate node, then click **Start**.
3. In the Node Detail panel, select **Shared Resources**.
4. Enable the corresponding shared resource and install it.

Task C Deploy and Start the SA Project

1. Select the SA Project in TIBCO ActiveMatrix Administrator, and then click **Deploy**.
2. Ensure that the status of the SA project is set to `deployed`. Select the SA project, and then click **Start**.

Task D Run the Process Included in the TIBCO Desginer Project

1. Open the `CustomerSubBW` project in TIBCO Desginer.
2. Select the `sub` process in the project panel.
3. Click the **Tester** tab in the project panel.
4. Click the **Start testing viewed process** button to open the Select Process to Load dialog.
5. Ensure that the `sub` process is selected.
6. Click **Load Selected**.

Expected Results

Enter or change the values for CUSTID, NAME, ADDRESS, CITY, STATE, ZIP, or PHONE in the Get_input.xml file.

The TIBCO ActiveMatrix BusinessWorks is triggered. The subscriber gets the data and invokes the NEWCUST Tuxedo service.

The NEWCUST service takes the details and writes them to the CustomerDetails file in the following format:

```
15: Arundhati: Vijaya Enclave: Bangalore: Karnataka: 560098:
1234567
```

The CustomerDetails file is located in the directory where you boot the Tuxedo service.

Chapter 5 EAR TO SA Example

Topics

Example Description

This example demonstrates how to convert an EAR file generated in TIBCO Desginer to a service assembly archive in TIBCO Business Studio, and then run it in TIBCO ActiveMatrix Administrator.

Location of the Example

The projects and files used by this example are located in the `Ear2Sa` folder provided in the `Tuxedo_Adapter_Service_Engine_Samples.zip` file.

Table 7 lists the projects and files used by this example.

Table 7 The Projects and Files Used by the Example

Project or File Name	Description
AMX Project	
`PurchaseOrderSOAProject`	An SOA project containing the service assembly archive (`PurchaseOrder.zip`) for this example. The `PurchaseOrder.zip` file is located in the `Deployment Packages` folder and can be deployed directly in TIBCO ActiveMatrix Administrator.
TIBCO Desginer Project	
`PurchaseOrder`	A TIBCO Desginer project containing the configuration information for the adapter instance and publisher used by the example. It also contains a pre-configured TIBCO ActiveMatrix BusinessWorks process to process the request from the publisher.
Tuxedo Service Codes	
`PODetails.h`	A Tuxedo header file containing the FML32 definition.
`adapteragent.h`	A file required to compile the Tuxedo server with the adapter agent. It is a copy of *TIBCO_HOME*`\adapter\`*version_number*`\include\adapteragent.h`
`client.c`	A source file for the Tuxedo client and is used to invoke the `POINFO` Tuxedo service.
`Get_output.xml`	An output XML file.
`Get_input_renderXsd.xsd`	A XSD file with XML schema.

Table 7 The Projects and Files Used by the Example

Project or File Name	Description
`inputdata.txt`	A file containing the input data in XML format.
`make.bat`	A file containing the commands required to get the Tuxedo service running on Windows. It sets the required environment variables, creates executables, loads the `ubbconfig` file, and boots the Tuxedo service.
`make.sh`	A file containing the commands required to get the Tuxedo service running on UNIX. It sets the required environment variables, creates executables, loads the `ubbconfig` file, and boots the Tuxedo service. This script should be run in a k-shell.
`server.c`	A source file for Tuxedo Server containing the `POINFO` Tuxedo service.
`ubbagent`	An `ubbconfig` file

Setting up the Example

To set up the environment for the example, complete the following tasks

- Task A, Prepare Tuxedo
- Task B, Configure the Environment for TIBCO ActiveMatrix
- Task C, Configure a TIBCO Runtime Agent file

To directly deploy the service assembly archive (`PurchaseOrder.zip`) provided with this example, refer to Running the Example. To create an EAR file for the `PurchaseOrder` project and convert the EAR file to a service assembly archive for this example, perform Task D through Task G.

- Task D, Create an EAR File
- Task E, Create an SOA Project
- Task F, Import the EAR file into TIBCO Business Studio
- Task G, Build a Service Assembly Archive

Task A Prepare Tuxedo

Refer to Preparing Tuxedo for detailed information about preparing Tuxedo for the example.

Task B Configure the Environment for TIBCO ActiveMatrix

For detailed information about configuring the Environment for TIBCO ActiveMatrix, refer to Configuring the Environment for TIBCO ActiveMatrix.

Task C Configure a TIBCO Runtime Agent file

The adapter works as a Tuxedo Workstation client. You need to modify the `adtuxedo_wrkstn.tra` file. The file is located in the *TIBCO_HOME*`\adapter\adtuxedo\`*veriosn_number*`\bin` directory. At a minimum, modify:

- `tibco.env.useNewEclipseUI`— set it to `ON`.
- `tibco.env.TUXConnNumber`— the number of connections with the Tuxedo application. The default value is 10.
- `tibco.env.RVAgentThreadNumber`— the number of threads to be used by the Rendezvous agent-based publisher. The default value is 4.

- `tibco.env.WSNADDR`—The IP address of the WSL (Workstation Listener) that the client will contact, along with a free port number. This value must be reflected in the `ubbconfig` file.

Refer to *TIBCO ActiveMatrix BusinessWorks Service Engine Configuration and Deployment* for detailed information about TIBCO Runtime Agent file.

Task D Create an EAR File

To create an EAR file for the `PurchaseOrder` project, follow these steps:

1. Open the `PurchaseOrder` project in TIBCO Desginer.
2. Select a folder and find the Enterprise Archive resource in the Palette panel.
3. Drag and drop an Enterprise Archive resource into the Design panel.

   The archive is displayed in the design panel. In the Configuration tab, enter **`PurchaseOrder`** in the Name field and specify the EAR file location in the File Location field.
4. Double click **`PurchaseOrder.`**
5. Drag and drop an Adapter Archive resource from the Palette panel to the Design panel.

   In the Configuration tab, specify the Tuxedo adapter instance configured in the `PurchaseOrder` project and click **Apply**.
6. Drag and drop a Process Archive resource from the Palette panel to the Design panel.

   In the Process tab, specify the TIBCO ActiveMatrix BusinessWorks process that is the `Pub` process configured in the `PurchaseOrder` project and click **Apply**.
7. Select the **`PurchaseOrder`** archive in the Project panel and click the **Build Archive** button in the Configuration tab to create the `PurchaseOrder.ear`.

Task E Create an SOA Project

1. Select **File > New > Project...** from the Menu to open the New Project dialog.
2. Expand **TIBCO ActiveMatrix > ActiveMatrix SOA Project**, and then click **Next**.
3. Type **`PurchaseOrderSOAProject`** in the Project name field, and then click **Next**.
4. Click **Finish**.

Task F Import the EAR file into TIBCO Business Studio

1. Select **`PurchaseOrderSOAProject`** in the Project Explorer panel and select **Import...** from the pop-up menu to open the Import dialog.
2. Select **General** > **File System** and click **Next** to open the File System page in the Import dialog.
3. Specify the EAR file directory, where the EAR file created in Task D is located, in the From directory field.
4. Click **Finish**.

The `PurchaseOrder.ear` file is imported into the `PurchaseOrderSOAProject` project.

Task G Build a Service Assembly Archive

Right-click **`PurchaseOrder.ear`** and select **Service Assembly Archive** from the pop-up menu to create the `PurchaseOrder.zip` file.

The `PurchaseOrder.zip` file is ready to be deployed in TIBCO ActiveMatrix Administrator.

Running the Example

To run the example, you need to complete the following tasks:

- Task A, Deploy an SA File
- Task B, Start the Node
- Task C, Deploy and Start the SA Project

Task A Deploy an SA File

Before proceeding, ensure that the HSQLDB, Management Daemon, TIBCO ActiveMatrix Administrator, TIBCO ActiveMatrix Administrator Server, TIBCO Enterprise Message Service Server, and Tuxdo service are running.

To deploy the SA project (the ZIP file) in TIBCO ActiveMatrix Administrator:

1. Select the **Deploy to an Environment** perspective.
2. Choose the Environment, and then click the **Upload Service Assembly** button.
3. Enter a name for the SA in the Name field.
4. Click the **Browse** button to select the archived ZIP file (`PurchaseOrder.zip`).
5. Check the **Import Shared Resource Definitions** checkbox to create the shared resources definitions.
6. Click **OK**.
7. In the Service Assemblies panel, select the service assembly.
8. Click the **Service Units** button.
9. For each service unit in the service assembly, follow these steps:
    a. Select a service unit in the Service Units table.
    b. In the Node Mapping tab, select the nodes which you want to deploy the service unit to from the Available Nodes list, and then click the right arrow button. The nodes will appear in the Mapped Nodes list.
    c. Click **Save**.

Task B Start the Node

1. In the web browser, select **Configure an Environment** in the Perspective drop-down list, and then select the appropriate Environment.

2. In the Nodes table, select the appropriate node, then click **Start**.

Task C Deploy and Start the SA Project

1. Select the SA project in TIBCO ActiveMatrix Administrator, and then click **Deploy**.
2. Ensure that the status of the SA project is set to `deployed`. Select the SA project, and then click **Start**.

Expected Results

In a command window, start Tuxedo Client and specify the `Order ID`. For example, specify the `Order ID` as follows:

your_directory`\Ear2Sa\Tuxedo Service Codes\PurchaseOrder>clt 1000`

After sending the request from Tuxedo Client, the `POINFO` Tuxedo service is invoked. The Tuxedo service puts the data into the JMS server. The adapter gets the data from the JMS server and publishes to the TIBCO environment. The TIBCO ActiveMatrix BusinessWorks process receives the data and writes it in an XML output file. For this example, the XML output file is the `Get_output.xml` file and is located in the *TIBCO_HOME*`\adapter\adtuxedo\`*version_number*`\examples\AgentBased\PurchaseOrder` directory.

Index

U