BEA WebLogic Integration™

BEA WebLogic Integration Migration Guide

Release 7.0
Document Date: June 2002

BEA WebLogic Integration Migration Guide

Part Number	Date	Software Version
N/A	June 2002	7.0

Contents

About This Document

BEA WebLogic Integration Migration Guide is organized as follows:

- Chapter 1, "Introduction to Migration," provides background information helpful to read before migrating to WebLogic Integration 7.0.
- Chapter 2, "Migrating WebLogic Integration 2.1 to WebLogic Integration 7.0," provides procedures for migrating BEA WebLogic Integration 2.1 or BEA WebLogic Integration 2.1 Service Pack 1 (SP1) to BEA WebLogic Integration 7.0.
- Chapter 3, "Other Migration Topics," provides information about WebLogic Server trusted relationships and about migrating a clustered WebLogic Integration application. It also provides procedures for configuring Application Integration Adapter EAR files.

What You Need to Know

BEA WebLogic Integration Migration Guide is designed for WebLogic Integration users who want to migrate to WebLogic Integration 7.0.

e-docs Web Site

BEA product documentation is available on the BEA corporate Web site. From the BEA Home page, click on Product Documentation or go directly to the "e-docs" Product Documentation page at `http://edocs.bea.com`.

How to Print the Document

You can print a copy of this document from a Web browser, one file at a time, by using the File→Print option on your Web browser.

A PDF version of this document is available on the WebLogic Integration documentation Home page on the e-docs Web site (and also on the documentation CD). You can open the PDF in Adobe Acrobat Reader and print the entire document (or a portion of it) in book format. To access the PDFs, open the WebLogic Integration documentation Home page, click the PDF files button and select the document you want to print.

If you do not have the Adobe Acrobat Reader, you can get it for free from the Adobe Web site at `http://www.adobe.com/`.

Related Information

The following resources are also available:

- BEA WebLogic Server 7.0 documentation (`http://edocs.bea.com/wls/docs70/index.html`)
- BEA WebLogic Integration 2.1 documentation (`http://edocs.bea.com/wlintegration/v2_1sp/index.html`)
- BEA WebLogic Integration 7.0 documentation (`http://edocs.bea.com/wli/docs70/index.html`)

Contact Us!

Your feedback on the WebLogic Integration documentation is important to us. Send us e-mail at **docsupport@beasys.com** if you have questions or comments. Your comments will be reviewed directly by the BEA professionals who create and update the WebLogic Integration documentation.

In your e-mail message, please indicate that you are using the documentation for this release of WebLogic Integration.

If you have any questions about this version of WebLogic Integration, or if you have problems installing and running WebLogic Integration, contact BEA Customer Support through BEA WebSUPPORT at **www.beasys.com**. You can also contact Customer Support by using the contact information provided on the Customer Support Card, which is included in the product package.

When contacting Customer Support, be prepared to provide the following information:

- Your name, e-mail address, phone number, and fax number
- Your company name and company address
- Your machine type and authorization codes
- The name and version of the product you are using
- A description of the problem and the content of pertinent error messages

Documentation Conventions

The following documentation conventions are used throughout this document.

Convention	Item
Ctrl+Tab	Indicates that you must press two or more keys simultaneously.
italics	Indicates emphasis or book titles.

Convention	Item

`monospace text`	Indicates code samples, commands and their options, data structures and their members, data types, directories, and file names and their extensions. Monospace text also indicates text that you must enter from the keyboard.
	Examples:
	`#include <iostream.h> void main ( ) the pointer psz`
	`chmod u+w *`
	`\tux\data\ap`
	`.doc`
	`tux.doc`
	`BITMAP`
	`float`

`monospace boldface text`	Identifies significant words in code. *Example*: `void` **`commit`** `( )`
`monospace italic text`	Identifies variables in code. *Example*: `String` *`expr`*
UPPERCASE TEXT	Indicates device names, environment variables, and logical operators. *Examples*: LPT1 SIGNON OR
{ }	Indicates a set of choices in a syntax line. The braces themselves should never be typed.

`[ ]`	Indicates optional items in a syntax line. The brackets themselves should never be typed.
	Example:
	`buildobjclient [-v] [-o name ] [-f` *`file-list`*`]...`
	`[-l` *`file-list`*`]...`

|	Separates mutually exclusive choices in a syntax line. The symbol itself should never be typed.

Convention	Item

`...`	Indicates one of the following in a command line:
	■ That an argument can be repeated several times in a command line
	■ That the statement omits additional optional arguments
	■ That you can enter additional parameters, values, or other information
	The ellipsis itself should never be typed.
	Example:
	`buildobjclient [-v] [-o name ] [-f file-list]...`
	`[-l file-list]...`

. . .	Indicates the omission of items from a code example or from a syntax line. The vertical ellipsis itself should never be typed.

CHAPTER

1 Introduction to Migration

This document provides the procedures you must complete to migrate WebLogic Integration 2.1 or WebLogic Integration 2.1 Service Pack 1 (SP1) to WebLogic Integration 7.0.

This section provides information about the following topics:

- Why Migrate?
- Migrating from Releases Prior to WebLogic Integration 2.1
- Adding WebLogic Integration to a WebLogic Server or WebLogic Portal Application

Why Migrate?

Migration is required because of the following differences between WebLogic Integration 7.0 and WebLogic Integration 2.1 or WebLogic Integration 2.1 Service Pack 1 (SP1):

- New database schema in WebLogic Integration 7.0
- RosettaNet workflow changes
- Security keystore changes
- Clustering changes
- Changes made between Releases 6.1 and 7.0 of WebLogic Server:
    - Startup scripts
    - Directory structure
    - Java Messaging Services
    - Enterprise JavaBeans (EJB) 2.0
    - Servlets
    - Thread pool size
    - Web applications
    - WebLogic Server clusters
    - Apache Xerces XML parser
    - Apache Xalan XML transformer
    - Security

Note: These differences are important during a migration to WebLogic Integration 7.0 because WebLogic Integration 7.0 is a software framework and a set of services built on top of WebLogic Server 7.0.

For more information about WebLogic Server 7.0 changes, see “Upgrading WebLogic Server 6.x to Version 7.0” in *BEA WebLogic Server Upgrade Guide*. This document is available, in the BEA WebLogic Server documentation set, at the following URL:

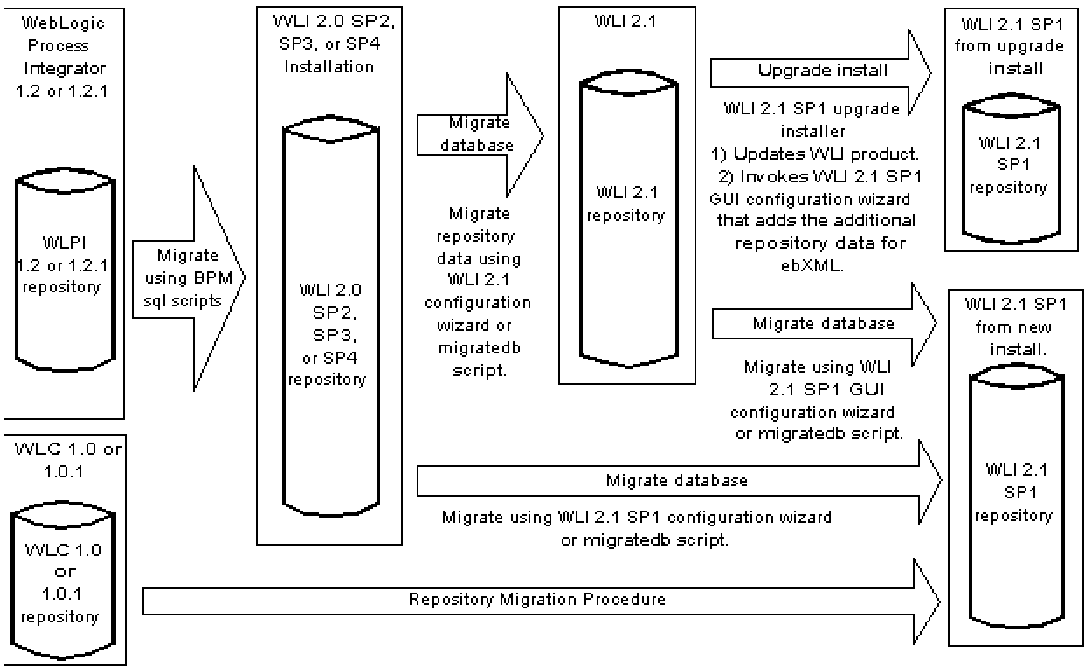
http://e-docs.bea.com/wls/docs70/upgrade/index.html

Note: As used in this document, the terms *upgrading* and *migrating* are synonymous. The term *migrating* refers to an upgrade from an older release to a newer one. Do not confuse this upgrade process with that of moving clusterable services from one instance of WebLogic Server to another.

Migrating from Releases Prior to WebLogic Integration 2.1

If you are migrating to WebLogic Integration 7.0 from a release prior to WebLogic Integration 2.1 Service Pack 1 (SP1), you must first update to WebLogic Integration 2.1 SP1, as shown in the following figure.

Figure 1-1 Migration Paths to WebLogic Integration 2.1 Service Pack 1


WebLogic Process Integrator 1.2 or 1.2.1
WLPI 1.2 or 1.2.1 repository
Migrate using BPM sql scripts
WLI 2.0 SP2, SP3, or SP4 Installation
WLI 2.0 SP2, SP3, or SP4 repository
Migrate database
Migrate repository data using WLI 2.1 configuration wizard or migratedb script.
WLI 2.1
WLI 2.1 repository
Upgrade install
WLI 2.1 SP1 upgrade installer 1) Updates WLI product. 2) Invokes WLI 2.1 SP1 GUI configuration wizard that adds the additional repository data for ebXML.
WLI 2.1 SP1 from upgrade install
WLI 2.1 SP1 repository
Migrate database
Migrate using WLI 2.1 SP1 GUI configuration wizard or migratedb script.
WLI 2.1 SP1 from new install.
WLI 2.1 SP1 repository
Migrate database
Migrate using WLI 2.1 SP1 configuration wizard or migratedb script.
WLC 1.0 or 1.0.1
WLC 1.0 or 1.0.1 repository
Repository Migration Procedure

After migrating to WebLogic Integration 2.1 SP1, you can then follow the instructions in this guide for migrating from WebLogic Integration 2.1 SP1 to WebLogic Integration 7.0.

Note: If you are migrating from WebLogic Integration 2.0 to WebLogic Integration 7.0, we recommend that you migrate using the following two-step procedure: 1) migrate from WebLogic Integration 2.0 to WebLogic Integration 2.1 SP1, 2) migrate from WebLogic Integration 2.1 SP1 to WebLogic Integration 7.0. This migration procedure is preferred because it minimizes the number of migration steps. Therefore we recommend using this two-step procedure instead of the following three-step procedure: 1) migrate from WebLogic Integration 2.0 to WebLogic Integration 2.1, 2) migrate from WebLogic Integration 2.1 to WebLogic Integration 2.1 SP1, 3) migrate from WebLogic Integration 2.1 SP1 to WebLogic Integration 7.0.

For information about migrating from earlier releases to WebLogic Integration 2.1 SP1, see *Migrating to BEA WebLogic Integration Release 2.1* at the following URL:

`http://edocs.bea.com/wlintegration/v2_1sp/migrate/index.htm`

Adding WebLogic Integration to a WebLogic Server or WebLogic Portal Application

The procedure provided in Chapter 2, "Migrating WebLogic Integration 2.1 to WebLogic Integration 7.0," describes the steps for migrating from WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 to WebLogic Integration 7.0. It does not, however, explain how to add WebLogic Integration to an existing application based on a pre-release 7.0 version of either WebLogic Server or WebLogic Portal. If you are adding WebLogic Integration to a WebLogic Server or WebLogic Portal application, there are security realm differences that may affect this addition. For more information, see *Introduction to WebLogic Platform 7.0 Security* in the BEA WebLogic Platform document set, available at the following URL:

`http://edocs.bea.com/platform/docs70/secintro/index.html`

CHAPTER

2 Migrating WebLogic Integration 2.1 to WebLogic Integration 7.0

This section provides the procedure for migrating from BEA WebLogic Integration 2.1 or BEA WebLogic Integration 2.1 Service Pack 1 (SP1) to BEA WebLogic Integration 7.0.

It includes the following steps:

- Step 1. Retrieve Your WebLogic Integration 2.1 Database Connection Information
- Step 2. Stop Your WebLogic Integration 2.1 Application
- Step 3. Back Up Your Application
- Step 4. Install WebLogic Integration 7.0
- Step 5. Create a New Domain and Configure Database Information
- Step 6. Migrate Your Database
- Step 7. Migrate Components of Your WebLogic Integration Application
- Step 8. Migrate Your Security Realm Data
- Step 9. Migrate to the Keystore for Security

- Step 10. Migrate Your RosettaNet Workflows
- Step 11. Deploy Your Application Views
- Step 12. Start and Test Your WebLogic Integration Application

Step 1. Retrieve Your WebLogic Integration 2.1 Database Connection Information

Later in this procedure ("Step 5. Create a New Domain and Configure Database Information" on page 2-6) you need to use information about the database configuration of your WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 installation. Retrieve that information now and note it for later.

To retrieve this information, complete the following procedure:

1. Start the WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 Database Wizard by completing the procedure appropriate for your platform:

   Windows:

   a. Select the domain that contains WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 database information.

   b. Start the Database Wizard for the appropriate domain.

For this domain . . .	Choose . . .
`wlidomain`	Start→BEA WebLogic E-Business Platform→WebLogic Integration 2.1→Configure
`samples`	Start→BEA WebLogic E-Business Platform→WebLogic Integration 2.1→Samples→Configure
`eaidomain`	Start→BEA WebLogic E-Business Platform→WebLogic Integration 2.1→Additional Preconfigured Domains→EAI Domain→Configure

For this domain . . .	Choose . . .
`bpmdomain`	Start→BEA WebLogic E-Business Platform→WebLogic Integration 2.1→Additional Preconfigured Domains→BPM Domain→Configure

The Choose Configuration Option dialog box is displayed.

UNIX:

a. Execute the following commands:

```
cd WLI_HOME/bin
wliconfig
```

The Choose BEA Home Directory dialog box is displayed.

b. Select an existing BEA Home directory, and then click Next.

The Choose Domain to Configure dialog box is displayed.

c. Select a domain, and then click Next.

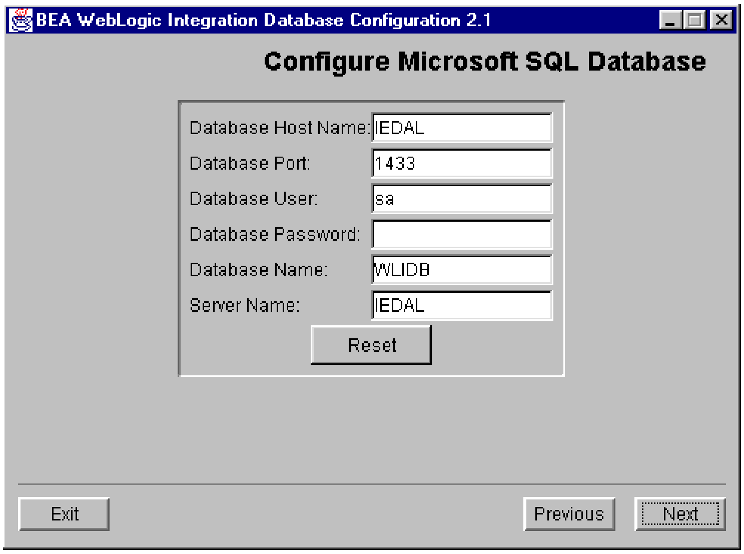
The Choose Configuration Option dialog box is displayed.

2. Select Switch Database and click Next.

   The Select Database dialog box is displayed.

3. Select a database type (Oracle, Microsoft SQL Server, or Sybase) and select Next.

   The Configure *`database_type`* Database dialog box is displayed. The name of the database type you selected replaces *`database_type`* in the window title. In this example Microsoft SQL is selected, so the Configure Microsoft SQL Database dialog box is displayed.



Configure Microsoft SQL Database

Database Host Name:	IEDAL
Database Port:	1433
Database User:	sa
Database Password:	
Database Name:	WLIDB
Server Name:	IEDAL

4. Record the database configuration information listed in the fields. This information will be required in "Step 5. Create a New Domain and Configure Database Information" on page 2-6.

5. Click Exit.

Step 2. Stop Your WebLogic Integration 2.1 Application

Before you migrate to WebLogic Integration 7.0, stop your WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 application. Make sure that:

- Your WebLogic Integration application is in a quiescent state.
- All instances of WebLogic Server in the application are shut down.
- No messages are being sent.

Step 3. Back Up Your Application

Before you attempt to migrate to WebLogic Integration 7.0, we strongly recommend that you back up the following:

- Your entire database
- Directories and files that contain your WebLogic Integration application including the following:
  - `config.xml` file
  - Certificates and private keys (if used)
  - Classes and JAR files used in business operations (if used)
- Security realm data

We also recommend that you export all your WebLogic Integration repository information and workflows:

- For instructions on exporting WebLogic Integration repository information, see "Exporting Repository Data" in "Configuring B2B Integration" in *Online Help for the WebLogic Integration B2B Console*. When selecting repository entities for export, select `All` in the Scope of Export field.
- For instructions on exporting your WebLogic Integration workflows through the WebLogic Integration Studio, see "Importing and Exporting Workflow Packages" in *Using the WebLogic Integration Studio*.

Step 4. Install WebLogic Integration 7.0

Install WebLogic Integration 7.0 in one of two modes:

- Custom Installation — When you install in this mode, and select the WebLogic Integration component, the installer automatically installs the WebLogic Server component, too.

- Typical Installation — When you install in this mode, the installer automatically installs WebLogic Server, WebLogic Integration, WebLogic Portal, and WebLogic Workshop components by default.

For instructions, see *Installing BEA WebLogic Platform* in the BEA WebLogic Platform document set, available at the following URL:

`http://edocs.bea.com/platform/docs70/install/index.html`

Warning: Do not create a new WebLogic Integration repository or database after the installation is complete. Do not run the `RunSamples` script.

Step 5. Create a New Domain and Configure Database Information

The directory structure for WebLogic Server domains was changed between Releases 6.x and 7.0. This change affects many scripts and settings of environment variables. For example, changes in the directory structure affect the settings of the `classpath` and `path` environment variables and the script that starts WebLogic Integration (`startWebLogic`). For this reason, this procedure provides steps for creating a new WebLogic Integration 7.0 domain and then moving your WebLogic Integration application-specific components to it, as described in "Step 7. Migrate Components of Your WebLogic Integration Application" on page 2-14.

The following procedure allows you to create a new domain for a single-server (standalone) configuration. For instructions on creating a multiple-server configuration, see "Step 2. Create a WebLogic Integration Domain" in "Configuring a Clustered Deployment" in *Deploying BEA WebLogic Integration Solutions*.

To create and customize a new domain for a single server:

1. Run the Configuration Wizard by completing the procedure appropriate for your platform:

- On a Windows system you have a choice of three methods:
  - To start the Configuration Wizard from a menu:

    Choose Start→Programs→BEA WebLogic Platform 7.0→Domain Configuration Wizard.
  - To start the Configuration Wizard in graphical mode from the command line:

    ```
    cd c:\bea\weblogic700\integration
    setenv.cmd
    cd c:\bea\weblogic700\common\bin
    dmwiz
    ```
  - To start the Configuration Wizard in console or text based mode:

    ```
    cd c:\bea\weblogic700\integration
    setenv.cmd
    cd c:\bea\weblogic700\common\bin
    dmwiz console
    ```
- On a UNIX system you have a choice of two methods:
  - To start the Configuration Wizard in graphical mode:

    ```
    cd /home/joe/bea/weblogic700/integration
    . setenv.sh
    cd /home/joe/bea/weblogic700/common/bin
    dmwiz.sh
    ```
  - To start the Configuration Wizard in console or text based mode:

    ```
    cd /home/joe/bea/weblogic700/integration
    . setenv.sh
    cd /home/joe/bea/weblogic700/common/bin
    dmwiz.sh console
    ```

  The Choose Domain Type and Name window is displayed.

  For more information, see *Using the Configuration Wizard* in the BEA WebLogic Platform document set, available at the following URL:

  `http://edocs.bea.com/platform/docs70/confgwiz/index.html`

2. Pick a domain template that contains the components of WebLogic Integration required for your application.

If your WebLogic Integration application uses the following WebLogic Integration features . . .	Choose the following domain template . . .
Application integration, data integration, business process management (BPM), and B2B integration	WLI (WebLogic Integration) Domain
Application integration, data integration, and business process management (BPM)	EAI (Enterprise Application Integration) Domain
Business process management (BPM)	BPM (Business Process Management) Domain

For more information about these templates, see "WebLogic Integration Configuration Templates" in "Getting Started" in *Starting, Stopping, and Customizing BEA WebLogic Integration* and *Configuration Wizard Template Reference* in the BEA WebLogic Platform document set, available at the following URL:

`http://edocs.bea.com/platform/docs70/template/index.html`

Warning: Make sure you select a WebLogic Integration template for creating the new domain; do not use a WebLogic Server or a WebLogic Portal template. Specifying a WebLogic Integration template ensures that the domain created in this step is based on the WebLogic Server 6.x security realm in compatibility mode. The new WebLogic Server 7.0 realm, based on LDAP, is not supported with this release of WebLogic Integration. If you create a new domain by selecting a WebLogic Server template, the new domain uses the new WebLogic Server 7.0 security realm, which is based on LDAP.

Note: It is recommended that you choose the same domain type used with your WebLogic Integration 2.1 application. For example, if you used the preconfigured EAI Domain for your WebLogic Integration 2.1 application, choose the EAI Domain template in this step.

3. In the Choose Server Type dialog box, select Single Server (Standalone Server).
4. Continue to provide the information required to create the new domain when prompted. When the process is complete, select End Configuration Wizard in the Configuration Wizard Complete dialog box, and then click the Done button to dismiss the Configuration Wizard.

5. (Optional) Save a copy of the generated `config.xml` file which contains configuration information, embedded in comments. The comments are deleted when the instance of WebLogic server is started. Make a copy of the existing `config.xml` file by entering the commands appropriate for your operating system:

   - Windows:

     ```
     copy config.xml config.xml.backup
     ```

   - UNIX:

     ```
     cp config.xml config.xml.backup
     ```

6. Configure the database connection information for the new domain. Start the WebLogic Integration Database Wizard (`wliconfig`) in the new domain created in steps 1 to 4.

   For example, if you created a domain named `mydomain` in the default location, enter the commands appropriate for your operating system:

   - Windows:

     ```
     cd %BEA_HOME%\user_projects\mydomain
     wliconfig
     ```

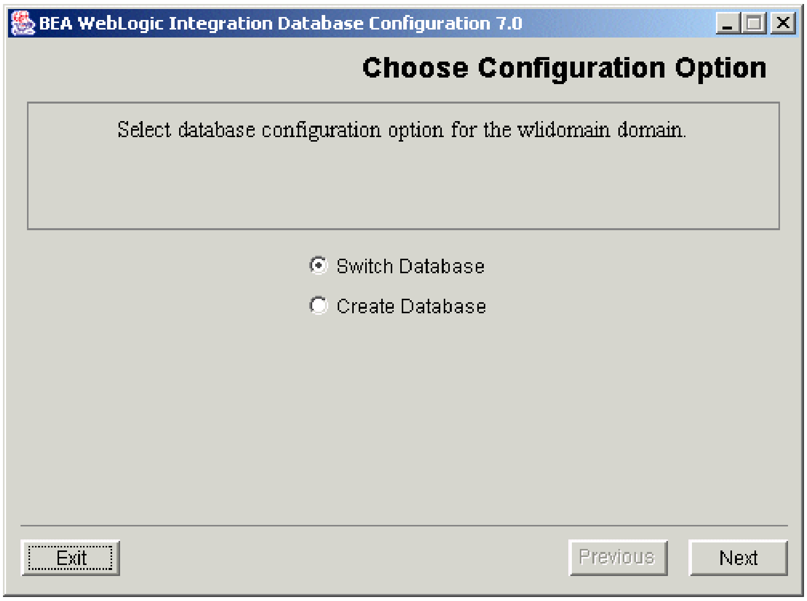
   - UNIX:

     ```
     cd $BEA_HOME/user_projects/mydomain
     wliconfig
     ```

   The Choose Configuration Option window is displayed.

7. Select the Switch Database option, as shown in the following figure, and click Next.

Figure 2-1 Choose Configuration Option


BEA WebLogic Integration Database Configuration 7.0

Choose Configuration Option

Select database configuration option for the wlidomain domain.

[✓ Switch Database]
Create Database

The Select Database window is displayed.

8. Select the same database type that you used for your WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 application.

   The Configure Database window is displayed.

9. Enter the database connection information that you noted in "Step 1. Retrieve Your WebLogic Integration 2.1 Database Connection Information" on page 2-2. Click Next.

   When the Database Wizard completes the configuration, the Changes Successful window is displayed.

10. Click Finish.

For information about running the Database Wizard, see "Using the Database Wizard" in "Customizing WebLogic Integration" in *Starting, Stopping, and Customizing BEA WebLogic Integration*.

Warning: Do not start the instance of WebLogic Server for this domain and do not create the database.

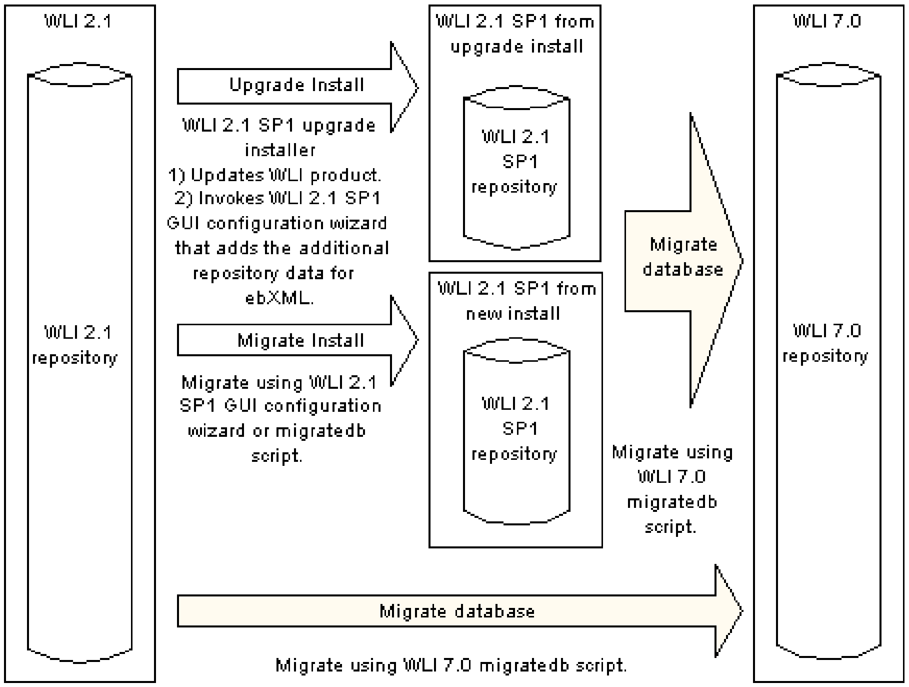
WebLogic Integration System Identities

In WebLogic Integration 2.1 and WebLogic Integration 2.1 SP1, two system user identities are used: `wlpisystem` and `wlcsystem`. In WebLogic Integration 7.0, these two system user identities are replaced by a single system user identity: `wlisystem`. When you create a new domain in WebLogic Integration (as described in "Customizing WebLogic Integration" in "Creating a New Domain" in *Starting, Stopping, and Customizing BEA WebLogic Integration*), `wlisystem` is the only the system user identity that is created.

Step 6. Migrate Your Database

This section provides a procedure for converting the database schema for either WebLogic Integration 2.1 or WebLogic Integration 2.1 Service Pack 1 (SP1) to the WebLogic Integration 7.0 format. This upgrade procedure is represented by the gray arrows labeled Migrate database in Figure 2-2.

Figure 2-2 WebLogic Integration 7.0 Database Migration


WLI 2.1
WLI 2.1 repository
Upgrade Install
WLI 2.1 SP1 upgrade installer
1) Updates WLI product.
2) Invokes WLI 2.1 SP1 GUI configuration wizard that adds the additional repository data for ebXML.
Migrate Install
Migrate using WLI 2.1 SP1 GUI configuration wizard or migratedb script.
WLI 2.1 SP1 from upgrade install
WLI 2.1 SP1 repository
WLI 2.1 SP1 from new install
WLI 2.1 SP1 repository
Migrate database
Migrate using WLI 7.0 migratedb script.
WLI 7.0
WLI 7.0 repository
Migrate database
Migrate using WLI 7.0 migratedb script.

Caution: This migration procedure updates a single repository. The migration of repository data from an existing database instance to a new instance is not supported.

To update the existing repository data to the WebLogic Integration 7.0 format, complete the following procedure:

1. Go to the WebLogic Integration home directory and set the top-level WebLogic Integration environment variables by running the `setenv` script. The commands you must run to perform these tasks depend on which platform you are using:

   - Windows:

     ```
     cd c:\bea\weblogic700\integration
     setEnv.cmd
     ```

- UNIX:

  ```
  cd /home/joe/bea/integration
  . setenv.sh
  ```

2. In a text editor, open the `*WLI_HOME*\dbscripts\migrate\SystemRepData.xml` file and add the line shown in bold in the following listing.

Listing 2-1 SystemRepData.xml

```
<?xml version="1.0"?>
<!DOCTYPE wlc SYSTEM "WLC.dtd">
<wlc
       system-password="wlisystem"
       ignore-wlc="true"
>
```

3. Go to the `bin` directory in your WebLogic Integration home directory. For example:
   - Windows:

     ```
     cd c:\bea\weblogic700\integration\bin
     ```
   - UNIX:

     ```
     cd /home/joe/bea/weblogic700/integration/bin
     ```
4. Run the `setdomain` script on the domain you created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6. For example:
   - Windows:

     ```
     setdomain c:\bea\user_projects\mydomain
     ```
   - UNIX:

     ```
     setdomain /home/joe/bea/user_projects/mydomain
     ```
5. Run the `switchdb` script for the database you configured in "Step 5. Create a New Domain and Configure Database Information" on page 2-6. For example:

   ```
   switchdb oracle
   ```

For a description of this command, including details about valid options for it, see "WebLogic Integration Commands" in *Starting, Stopping, and Customizing BEA WebLogic Integration*.

6. Run the migration script to update the repository data from WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 to the WebLogic Integration 7.0 domain:

```
migratedb
```

Step 7. Migrate Components of Your WebLogic Integration Application

In "Step 5. Create a New Domain and Configure Database Information" on page 2-6, you created a new WebLogic Integration domain. In this step, you migrate your WebLogic Server application-specific components from your WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 application to the new domain.

Note: As used here, the term *WebLogic Integration application* refers to an application that you have developed, based on WebLogic Integration. It should not be confused with the `WebLogic Application` application element in the `config.xml` file.

Examples of WebLogic Server application-specific components include the following:

- JMS queues
- Enterprise applications—An enterprise application may include any number of EJBs, Web applications, and Connector modules.
- Enterprise JavaBeans (EJB)
- Web applications
- Connectors

Warning: Migrate only those application-specific components that you developed for your application; do not migrate the components used by WebLogic Integration itself. For example, you should not migrate the JMS queues used by WebLogic Integration, but you do need to migrate the custom

JMS queues that you developed for your WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 application. The JMS queues used by WebLogic Integration are automatically included in the `config.xml` file that is generated when you create the new domain in "Step 5. Create a New Domain and Configure Database Information" on page 2-6.

Warning: Do not copy files (for example, `config.xml`) from your WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 domain directory to your WebLogic Integration 7.0 domain directory. Due to changes in the directory structure of WebLogic Server 7.0 domains, WebLogic Server 6.x files are not compatible with WebLogic Server 7.0.

For instructions on migrating these WebLogic Server components and more information about migrating from WebLogic Server 6.x to WebLogic Server 7.0, see "Upgrading WebLogic Server 6.x to Version 7.0" in *BEA WebLogic Server Upgrade Guide*. This document is available, in the BEA WebLogic Server documentation set, at the following URL:

`http://e-docs.bea.com/wls/docs70/upgrade/index.html`

Note: As used in this document, the terms *upgrade* and *migrate* are synonymous. The term *migrate* refers to the process of changing your software from an older release to a newer one. Do not confuse this upgrade process with that of moving clusterable services from one WebLogic Server to another.

Because WebLogic Integration 7.0 is built on top of WebLogic Server 7.0, the changes to your application that are required for migrating from WebLogic Server 6.x to WebLogic Server 7.0 also affect any migration to WebLogic Integration 7.0.

Warning: WebLogic Server 7.0 supports a new security realm based on LDAP. WebLogic Integration 7.0, however, does not support the new security realm based on the LDAP. WebLogic Integration 7.0 supports the Compatibility realm which, in turn, supports both the File and RDBMS realms.

Application Integration

If your application invokes the application integration functionality provided by WebLogic Integration, you must configure your application integration adapter EAR file(s). For instructions, see "Configuring Application Integration Adapter EAR Files" on page 3-3.

Application-Specific JAR Files

If you create application-specific JAR files for your WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 application, you must copy those JAR files into the `WLI_HOME/lib` directory (where `WLI_HOME` represents the location of the WebLogic Integration 7.0 home directory).

Warning: Do not copy application integration JAR files from your WebLogic Integration 2.1 or 2.1 SP1 application to your WebLogic Integration 7.0 application. For more information, see "Application Integration CLASSPATH and Adapter Packaging Changes" on page 2-18.

Also, you must add an entry for your application-specific JAR to the `config.xml` and `application.xml` files for the WebLogic Integration 7.0 domain created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6.

The WebLogic Integration J2EE components are now deployed using the exploded form. To deploy J2EE components in the exploded form, you must use a new syntax when you specify a JAR file in the `config.xml` file. Do not cut and paste the application-specific JAR entries directly from your WebLogic Integration 2.1 or WebLogic Integration 2.1 `config.xml` file to your WebLogic Integration 7.0 `config.xml` file. Instead, follow the instructions in "Adding an EJB to the WLI Application Element" in "Customizing WebLogic Integration" in *Starting, Stopping, and Customizing BEA WebLogic Integration*.

startWebLogic Script

If custom changes have been made to the `startWebLogic` script that you run with your WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 application, you must make the same changes to the `startWebLogic` script for the WebLogic Integration 7.0 domain created in “Step 5. Create a New Domain and Configure Database Information” on page 2-6. For example, if you add an application-specific JAR file, containing business operations, to the `CLASSPATH` of the `startWebLogic` script, you must also add the same JAR file to the `CLASSPATH` of the WebLogic Integration 7.0 `startWebLogic` script (`startWebLogic.cmd` for Windows systems and `startWebLogic.sh` for UNIX systems).

For more information, see “Adding Java Classes to the CLASSPATH” in “Customizing WebLogic Integration” in *Starting, Stopping, and Customizing BEA WebLogic Integration*.

setenv Script

If the `setenv` script for your WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 application is customized, you must customize, in the same way, the `setenv` script for the WebLogic Integration 7.0 domain created in “Step 5. Create a New Domain and Configure Database Information” on page 2-6. For example, if you add an application-specific JAR file containing business operations to the `CLASSPATH` of the `setenv` script, you must add the same JAR file to the `CLASSPATH` of the WebLogic Integration 7.0 `setenv` script. The name of this script is `setenv.cmd` on Windows systems and `setenv.sh` on UNIX systems.

For more information, see “Adding Java Classes to the CLASSPATH” in “Customizing WebLogic Integration” in *Starting, Stopping, and Customizing BEA WebLogic Integration*.

Application Integration CLASSPATH and Adapter Packaging Changes

In WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 applications, you must include adapter Java classes in the system `CLASSPATH` for your instance of WebLogic Server. In WebLogic Integration 7.0 applications, however, adapter Java classes must be packaged in a single, self-contained application enterprise archive (EAR) file. Do not move any adapter Java classes or JAR files to your WebLogic Integration 7.0 installation, and do not add the adapter classes to the WebLogic Integration `CLASSPATH`. For instructions on configuring adapter EAR files, see “Configuring Application Integration Adapter EAR Files” on page 3-3.

B2B Transport Servlet

If your WebLogic Integration 7.0 application uses the business-to-business integration (B2B) functionality provided by WebLogic Integration, you must add an entry for the `TransportServletFilter` to the deployment descriptor for the default Web application. If, when you created a domain (see “Step 5. Create a New Domain and Configure Database Information” on page 2-6), you used a template that included B2B functionality, then your domain already has an entry for the `TransportServletFilter`.

For example, if you created a domain with the `WLI` domain template, an entry for `TransportServletFilter` is added to the `*DOMAIN_HOME*\applications\DefaultWebApp_myserver\WEB-INF\web.xml` file, where `*DOMAIN_HOME*` represents the pathname of the domain you created, as shown in the following listing.

Listing 2-2 Entry for TransportServerFilter in web.xml

```
<!-- WLI-B2Bi filter-begin.  DO NOT EDIT -->
<filter>
  <filter-name>TransportServletFilter</filter-name>
  <filter-class>com.bea.b2b.transport.http.TransportServletFilter</filter-class>
</filter>
<filter-mapping>
  <filter-name>TransportServletFilter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
<!-- WLI-B2Bi filter-end. -->
```

Step 8. Migrate Your Security Realm Data

If you add any data, such as entries for your own users or groups, to your WebLogic Integration 2.1 or 2.1 SP1 security realm, you must also add the same data to the new domain created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6.

Warning: Do not copy the `fileRealm.properties` and `SerializedSystemIni.dat` file from your WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 domain directory to your new WebLogic Integration 7.0 domain because the default system user identities and groups have changed. For more information, see "WebLogic Integration System Identities" on page 2-11.

Migrating from the RDBMS Realm

If you migrate from a WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 application in which the RDBMSRealm is used, you must change the realm specified, for the domain, in the `config.xml` file (created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6) from FileRealm to RDBMSRealm. Specifically, you must replace the Realm element in the `config.xml` file (shown in Listing 2-3) with the `RDBMSRealm`, `CachingRealm`, and `Realm` elements from your existing WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 `config.xml` file.

If you have not booted your instance of WebLogic Server, the generated `config.xml` file will contain the RDBMSRealm elements in a comment section of the file. You can remove the comment delimiters for this section, if you want to add the RDBMSRealm elements to your `config.xml` file. The default database, PointBase is already configured. You must update the database configuration information from your existing WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1`config.xml` file.

Listing 2-3 Realm Element to Be Replaced

```
<Realm CachingRealm="" FileRealm="myFileRealm" Name="myRealm"/>
```

Database attributes for your application should be configured already in your existing WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1`config.xml` file. Listing 2-4 shows an example listing for an Oracle database.

Listing 2-4 RDBMS Elements to Be Added

```
<RDBMSRealm DatabaseDriver="oracle.jdbc.driver.OracleDriver"
DatabaseURL="jdbc:oracle:thin:@(description=(address=(host=
MY_ORACLE_SERVER)(protocol=tcp)(port=1521))(connect_data=
(sid=MY_ORACLE_SID)))" DatabaseUserName="scott"
DatabasePassword="tiger" Name="wlpiRDBMSRealm"
RealmClassName="com.bea.wlpi.rdbmsrealm.RDBMSRealm"
SchemaProperties="getGroupNewStatement=true;
removeUserFromGroup=DELETE FROM USERMEMBER WHERE USERID
= ? AND GROUPID = ?;getAcls=SELECT NAME, PRINCIPAL,
PERMISSION FROM ACLENTRIES ORDER BY NAME,
PRINCIPAL;addUserToGroup=INSERT INTO USERMEMBER
(USERID, GROUPID) VALUES ( ?, ? );getGroupMembersUsers
=SELECT USERMEMBER.USERID, PASSWORD FROM USERMEMBER,
WLSUSER WHERE GROUPID = ? AND USERMEMBER.USERID =
WLSUSER.USERID;newGroup=INSERT INTO WLSGROUP (GROUPID)
VALUES ( ? );addGroupToGroup=INSERT INTO GROUPMEMBER
(GROUPMEMBERID, GROUPID) VALUES ( ?, ? );newUser=INSERT
INTO WLSUSER (USERID, PASSWORD) VALUES ( ? , ?
);removeGroupFromGroup=DELETE FROM GROUPMEMBER WHERE
GROUPMEMBERID = ? AND GROUPID = ?;deleteGroup4=DELETE FROM
WLSGROUP WHERE GROUPID = ?;deleteUser3=DELETE FROM WLSUSER
WHERE USERID = ?;deleteGroup3=DELETE FROM USERMEMBER WHERE
GROUPID = ?;getPermissions=SELECT DISTINCT PERMISSION
FROM ACLENTRIES;deleteUser2=DELETE FROM USERMEMBER WHERE
USERID = ?;getPermission=SELECT DISTINCT PERMISSION FROM
ACLENTRIES WHERE PERMISSION = ?;getUser=SELECT USERID,
PASSWORD FROM WLSUSER WHERE USERID = ?;deleteGroup2=DELETE
FROM ACLENTRIES WHERE PRINCIPAL = ?;deleteGroup1=DELETE FROM
GROUPMEMBER WHERE GROUPID = ?;deleteUser1=DELETE FROM
ACLENTRIES WHERE PRINCIPAL = ?;getAclEntries=SELECT
NAME, PRINCIPAL, PERMISSION FROM ACLENTRIES WHERE NAME = ?
ORDER BY PRINCIPAL;getGroupMembersGroups=SELECT GROUPMEMBERID,
GROUPID FROM GROUPMEMBER WHERE GROUPID = ?;getGroups=
SELECT GROUPID FROM WLSGROUP ORDER BY GROUPID;
getGroup=SELECT GROUPID FROM WLSGROUP WHERE GROUPID
= ?;getUsers=SELECT USERID, PASSWORD FROM WLSUSER
ORDER BY USERID"/>

<CachingRealm BasicRealm="wlpiRDBMSRealm"
CacheCaseSensitive="true" Name="wlpiCachingRealm"/>

<Realm CachingRealm="wlpiCachingRealm" FileRealm=
"myFileRealm" Name="myRealm"/>
```

Note: The `RDBMSRealm` element in Listing 2-4 must be one continuous line in your `config.xml` file. In Listing 2-4 the element is divided into many lines for readability.

Migration from the RDBMSRealm is supported only for the Microsoft SQL Server and Oracle databases.

Note: A new security model for trusted relationships is used in WebLogic Server 7.0. For more information, see “Trusted Relationships” on page 3-7.

Step 9. Migrate to the Keystore for Security

In WebLogic Integration 2.1 and WebLogic Integration 2.1 SP1 applications, private keys and the certificates associated with them are stored on the file system. WebLogic Platform 7.0 provides a keystore in which those keys and certificates can be stored. If your WebLogic Integration application is configured to communicate using the SSL protocol, or if it contains trading partners with delivery channel(s) that are configured to use the SSL protocol, message encyption, or digital signatures, we recommend that you convert your application to use the new keystore.

Use of the SSL protocol for communication between trading partners is supported for all WebLogic Integration B2B collaboration protocols (ebXML, XOCP, and RosettaNet). Message encryption and digital signatures, however, are supported only for RosettaNet 2.0.

Note: For WebLogic Integration 7.0, the only supported keystore provider is Java Keystore (JKS) from Sun Microsystems.

For more information about using a keystore with WebLogic Integration, see "Configuring the Keystore" in *Implementing Security with B2B Integration*.

Convert your WebLogic Integration application to use the new keystore by importing all existing certificates and private keys from the file system into the keystore. To perform this conversion, complete the following steps:

1. Go to the WebLogic Integration home directory and set the top-level WebLogic Integration environment variables by running the `setenv` script. The commands you must run to perform these tasks depend on which platform you are using:

   - Windows:

     ```
     cd c:\bea\weblogic700\integration
     setEnv.cmd
     ```

   - UNIX:

     ```
     cd /home/joe/bea/weblogic700/integration
     . setenv.sh
     ```

2. Go to the new WebLogic Integration 7.0 domain created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6 by completing the following command:

   ```
   cd DOMAIN_HOME
   ```

   `DOMAIN_HOME` represents the pathname of the new domain.

3. In a text editor, open the appropriate start script (`startWebLogic.cmd` script on Windows or `startWebLogic` script on UNIX) and make the changes described in the following steps:

   a. Find the line containing the `java` command that starts the instance of WebLogic Server, as shown in the following listing:

      ```
      %JAVA_HOME%\bin\java ... weblogic.Server
      ```

   b. Add the Java system property `wli.keystore.automigrate` to the `java` command line, and set the property equal to `true`, as shown (in bold) in the following listing:

      ```
      %JAVA_HOME%\bin\java ... -Dwli.keystore.automigrate=true
      weblogic.Server
      ```

   c. Add the Java system property `wli.keystore.password` to the `java` command line appropriate for your platform.

On a Windows platform, enter the string shown in bold in the following listing:

```
%JAVA_HOME%\bin\java ... -Dwli.keystore.automigrate=true
-Dwli.keystore.password=%KEYSTORE_PASS% weblogic.Server
```

On a UNIX platform, enter the string shown in bold in the following listing:

```
$JAVA_HOME/bin/java ... -Dwli.keystore.automigrate=true
-Dwli.keystore.password=$KEYSTORE_PASS weblogic.Server
```

In both listings, `KEYSTORE_PASS` represents the environment variable that contains the password of the keystore. We strongly recommend that you specify a password using an environment variable, as shown in the listings in this step, rather than including the passwords in the scripts. Passwords should not be stored in clear text in files.

The password specified in the `KEYSTORE_PASS` environment variable must match the password for the keystore that you specify in step 8.

d. Add the Java system property `wli.cakeystore.password` to the `java` command line appropriate for your platform.

On a Windows platform, enter the string shown in bold in the following listing:

```
%JAVA_HOME%\bin\java ... -Dwli.keystore.automigrate=true
-Dwli.keystore.password=%KEYSTORE_PASS%
-Dwli.cakeystore.password=%CAKEYSTORE_PASS% weblogic.Server
```

On a UNIX platform, enter the string shown in bold in the following listing:

```
$JAVA_HOME/bin/java ... -Dwli.keystore.automigrate=true
-Dwli.keystore.password=$KEYSTORE_PASS
-Dwli.cakeystore.password=$CAKEYSTORE_PASS weblogic.Server
```

In both listings, `CAKEYSTORE_PASS` represents the environment variable that contains the password of the root Certificate Authority (CA) keystore.

The password specified in the `CAKEYSTORE_PASS` environment variable must match the password for the root Certificate Authority (CA) that you specify in step 8.

e. For each security certificate used by your WebLogic Integration application, add the following system property to the `java` command line for your platform.

Windows:

```
-D Key.certificateName.password=%KEY_PASS1%
```

UNIX:

```
-D Key.certificateName.password=$KEY_PASS1
```

In both property settings, *`certificateName`* represents the Certificate Name and `KEY_PASS1` represents the environment variable that contains the password for the private key of the certificate. The settings for these properties should match the settings for these certificates in your WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 application.

Add this system property for all the security certificates used by your application. For example, if you specify two certificates, `cert1` and `cert2`, with the environment variables `passcert1` and `passcert2`, respectively (to define the private key passwords), the `java` command line in the `startWebLogic.cmd` script (for Windows) appears as shown in the following listing:

```
%JAVA_HOME%\bin\java ... -Dwli.keystore.automigrate=true
-Dwli.keystore.password=$KEYSTORE_PASS
-DKey.cert1.password=%passcert1%
-DKey.cert2.password=%passcert2% weblogic.Server
```

4. Set the environment variables for the private key passwords, the password for the keystore, and the password of the root Certificate Authority (CA) keystore that you defined in step 3.

   The values of the `KEYSTORE_PASS` and `CAKEYSTORE_PASS` environment variables must match the passwords for the keystore and the root Certificate Authority (CA) keystore that you specify in step 8.

5. In a text editor, open the `config.xml` file for the new domain (created in “Step 5. Create a New Domain and Configure Database Information” on page 2-6) and set `false` as the value of the `Deployed` attribute of the `WebLogic Integration` Application element, as shown in the following example.

   ```
   <Application Deployed="false" Name="WebLogic Integration"
   Path="c:/bea/weblogic700/integration/lib>” TwoPhase="true">
   ```

   When you set this attribute to `false`, the `WebLogic Integration` application (the collection of J2EE components that make up WebLogic Integration) is not deployed the next time the instance of the WebLogic Server for the new domain is booted.

6. Start your instance of WebLogic Server, using the new WebLogic Integration 7.0 domain you created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6, by completing the procedure appropriate for your platform.

- On a Windows system, you have a choice of two methods:

  - To start the server from a menu:

    Choose Start→BEA WebLogic Platform 7.0→User Projects→`domain`→Start Server.

  - To start the server from the command line:

    ```
    startWeblogic.cmd
    ```

- UNIX:

  ```
  startWebLogic
  ```

7. Log on to the WebLogic Server Administration Console:

   a. Open a new browser window.

   b. Enter the URL for your system's WebLogic Server Administration Console. The actual URL you enter depends on your system. It should conform to the following format:

   ```
   http://host:port/console
   ```

   The WebLogic Server Administration Logon page is displayed.

   c. Enter your WebLogic Server username and password, then click Sign In. The username and password entered here are the username and password that you specified when you created the domain in "Step 5. Create a New Domain and Configure Database Information" on page 2-6.

   The Welcome to BEA WebLogic Server Home page is displayed.

   Note: For more detailed instructions, see "Starting the WebLogic Server Administration Console" in "WebLogic Integration Administration and Design Tools" in *Starting, Stopping, and Customizing BEA WebLogic Integration*.

8. Create the keystores. For instructions, see "Creating the Keystores" in "Configuring the Keystore" in *Implementing Security with B2B Integration*.

9. Configure the WebLogic keystore provider for Java Keystore (JKS) from Sun Microsystems. For instructions, see "Configuring the WebLogic Keystore Provider" in "Configuring the Keystore" in *Implementing Security with B2B Integration*.

10. Shut down the WebLogic Server instance for the new WebLogic Integration 7.0 domain (created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6), by completing the procedure appropriate for your platform:

    - Windows:

      ```
      cd DOMAIN_HOME
      stopWeblogic.cmd
      ```

    - UNIX:

      ```
      cd DOMAIN_HOME
      stopWebLogic
      ```

    In both `cd` command lines, *`DOMAIN_HOME`* represents the pathname of the new domain you (which you created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6), and *`domain`* represents the directory you created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6.

11. In a text editor, open the `config.xml` file for the new domain (created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6) and set `true` as the value of the `Deployed` attribute of the `WebLogic Integration` Application element, as shown in the following example:

    ```
    <Application Deployed="true" Name="WebLogic Integration"
    Path="c:/bea/weblogic700/integration/lib>" TwoPhase="true">
    ```

    When you set this attribute to `true`, `WebLogic Integration` application (the collection of J2EE components that make up WebLogic Integration) is deployed the next time the instance of the WebLogic Server for the new domain is booted.

12. Start the instance of WebLogic Server using the new WebLogic Integration 7.0 domain (created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6) by completing the procedure appropriate for your platform:

- On a Windows system, you have a choice of two methods:
  - To start the server from a menu:

    Choose Start→BEA WebLogic Platform 7.0→User Projects→*domain*→Start Server.
  - To start the server from the command line:

    ```
    startWeblogic.cmd
    ```
- UNIX:

  ```
  startWebLogic
  ```

  When the instance of WebLogic Server boots, the B2B engine retrieves, from the repository, the locations of the certificates and the private keys associated with them. It then populates the keystore with those certificates and keys.

13. Shut down the WebLogic Server instance for the new WebLogic Integration 7.0 domain (created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6) by completing the procedure appropriate for your platform:
    - Windows:

      ```
      cd DOMAIN_HOME
      stopWeblogic.cmd
      ```
    - UNIX:

      ```
      cd DOMAIN_HOME
      stopWebLogic
      ```

    In both `cd` command lines, *`DOMAIN_HOME`* represents the pathname of the new domain that you created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6.

14. Remove the `wli.keystore.automigrate` system property from your `startWebLogic` script. This step is mandatory; it ensures that the next time the `startWebLogic` script is run, the B2B engine does not attempt to migrate the certificates and the private keys associated with them from the file system to the keystore, again. (This migration should be attempted only once.)

To remove this property, complete the following procedure:

a. In a text editor, open the start script for your system: `startWebLogic.cmd` for Windows or `startWebLogic` for UNIX.

b. Remove the only the `wli.keystore.automigrate` system property from your script.

Warning: Do not remove the `Key.`*`certificateName`*`.password`, the `wli.cakeystore.password`, and the `wli.keystore.password` system properties that you added in step 3.

If you make these changes to the same script you edited in step 3, your script appears as shown in the following listing.

Listing 2-5 Example startWebLogic.cmd Java Command for Windows

```
%JAVA_HOME%\bin\java ... -Dwli.keystore.password=%KEYSTORE_PASS%
-Dwli.cakeystore.password=%CAKEYSTORE_PASS% -DKey.cert1.password=%passcert1%
-DKey.cert2.password=%passcert2% weblogic.Server
```

Caution: This procedure should be done only once, to initially populate the keystore.

This procedure does not include steps for migrating the server certificate and the private key associated with it. For instructions on adding this server certificate and associated key to the keystore, see “Adding the Server Certificate Required by SSL” in “Configuring the Keystore” in *Implementing Security with B2B Integration*.

Step 10. Migrate Your RosettaNet Workflows

If your WebLogic Integration application includes workflows that implement the RosettaNet protocol, you must make changes to those workflows before running your application with WebLogic Integration 7.0.

Note: If your WebLogic Integration application uses RosettaNet workflows, see “RosettaNet Schema Changes” on page 3-11.

To make your RosettaNet workflows compatible with WebLogic Integration 7.0, complete the following procedure:

1. Go to the WebLogic Integration home directory and set the top-level WebLogic Integration environment variables. To set these variables, run the `setenv` script appropriate for your platform:

   - Windows:

     ```
     cd c:\bea\weblogic700\integration
     setEnv.cmd
     ```

   - UNIX:

     ```
     cd /home/joe/bea/weblogic700/integration
     . setenv.sh
     ```

2. Start an instance of WebLogic Server using the new WebLogic Integration 7.0 domain (which you created in “Step 5. Create a New Domain and Configure Database Information” on page 2-6) by completing the procedure appropriate for your platform.

   On a Windows system, you have a choice of two methods:

   - To start the server from a menu:

     Choose Start→BEA WebLogic Platform7.0→UserProjects→*`domain`*→*`servername`*.

   - To start the server from the command line:

     ```
     cd DOMAIN_HOME
     startWeblogic.cmd
     ```

   On a UNIX system:

   ```
   cd DOMAIN_HOME
   startWebLogic
   ```

   In both cases, *`DOMAIN_HOME`* represents the pathname of the new domain that you created in “Step 5. Create a New Domain and Configure Database Information.”

3. Start the WebLogic Integration Studio as described in “Starting the Studio” in “WebLogic Integration Administration and Design Tools” in *Starting, Stopping, and Customizing BEA WebLogic Integration*. If an instance of your RosettaNet workflow is currently stored in the WebLogic Integration repository, skip to step 5. (Instances of templates stored in the WebLogic Integration 2.1 repository are still available after completing “Step 6. Migrate Your Database” on page 2-11.)

4. Import your WebLogic Integration RosettaNet workflow template into the Studio by completing the following procedure:

   a. From the Studio menu bar, choose Tools→Import Package... .

   b. In the Import:Select File window, select the JAR file that contains the RosettaNet workflow templates to be converted for your application. Click Next.

   c. Expand the drop-down list under Target Organization and select the appropriate organization for your application.

   d. Click Import, and then Close.

   Note: For more detailed instructions on importing workflows, see "Importing and Exporting Workflow Packages" in *Using the WebLogic Integration Studio*.

5. Open your WebLogic Integration RosettaNet workflow template in the Studio by completing the following procedure:

   a. In the left pane, expand the drop-down list under Organization and select the appropriate organization for your application.

   b. In the left pane, expand the Templates folder. A list of all the templates for the application is displayed.

   c. Expand the template folder targeted for conversion in the left pane.

   d. Within the template folder, right-click the folder targeted for conversion. (You can identify it by the date and time stamp shown on it.) A menu is displayed.

   e. Select Open.

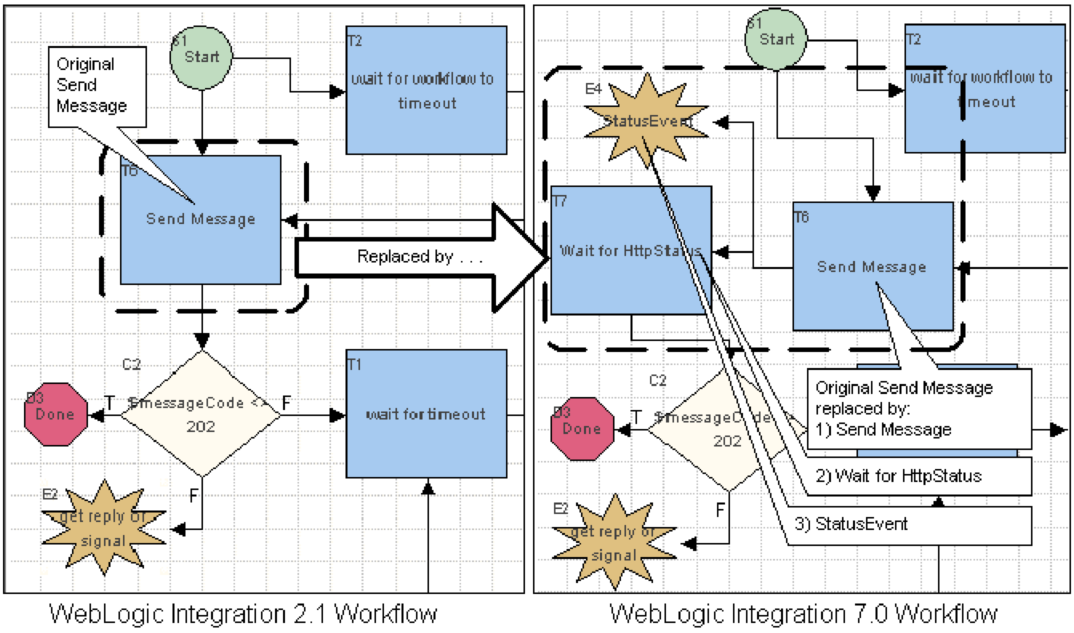
      The start, task, decision, and event nodes that make up the workflow are displayed.

6. In the workflow, replace each instance of a task node with a Send Business Message action with the following three nodes:

   - Send Business Message (task node)
   - RosettaNet Status Event (event node)
   - Wait for HTTP Status (task node)

Detailed instructions for adding these nodes are provided in steps 6-a to 6-v. The following figure shows how your workflow is changed when you substitute these nodes for all instances of a task node with a Send Business Message action.

Figure 2-3 Differences Between RosettaNet Workflows for WebLogic Integration 2.1 and WebLogic Integration 7.0



In RosettaNet workflows for WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1, the Send Business Message task sends the message, receives the HTTP status code, and then proceeds to the next node. In RosettaNet workflows for WebLogic Integration 7.0, however, the Send Business Message task sends the message, and an event node (RosettaNet Status Event) waits for the HTTP status. When it receives the status, the event node marks another task node (Wait for HTTP Status) as done, and the workflow proceeds from the Wait for HTTP Status task node.

In RosettaNet workflows for WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1, the Send Business Message task is synchronous: it waits for an HTTP status reply before proceeding to the next node in the workflow. In RosettaNet workflows for WebLogic Integration 7.0, the Send Business Message task is asynchronous: it does not wait for the HTTP status reply before proceeding to the next node in the workflow.

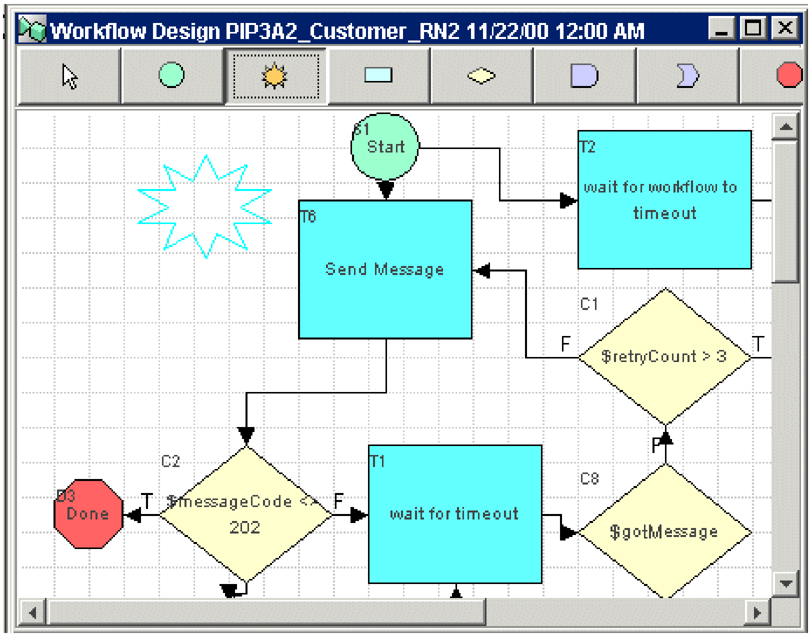
Note: The PIP3A2_Customer workflow shown in Figure 2-3 is taken from the B2B RosettaNet Security sample. The B2B RosettaNet Security sample has been converted to run on WebLogic Integration 7.0. The workflows for it are located in the following file:
`*SAMPLES_HOME*/integration/samples/RN2Security/workflow/RN2Workflows.jar`
In this pathname, `*SAMPLES_HOME*` represents the WebLogic Platform samples directory. The workflows available there provide examples of the Send Message changes described in this section. If you have not altered the original PIP3A2 workflows supplied with earlier releases of WebLogic Integration, you can replace the PIP3A2 workflows in your application with the new workflows supplied with the B2B RosettaNet Security sample.

To make an instance of the Send Business Message task in a RosettaNet workflow compatible with WebLogic Integration 7.0, complete the following steps:

a. In the workflow, find the Send Business Action task to be converted.

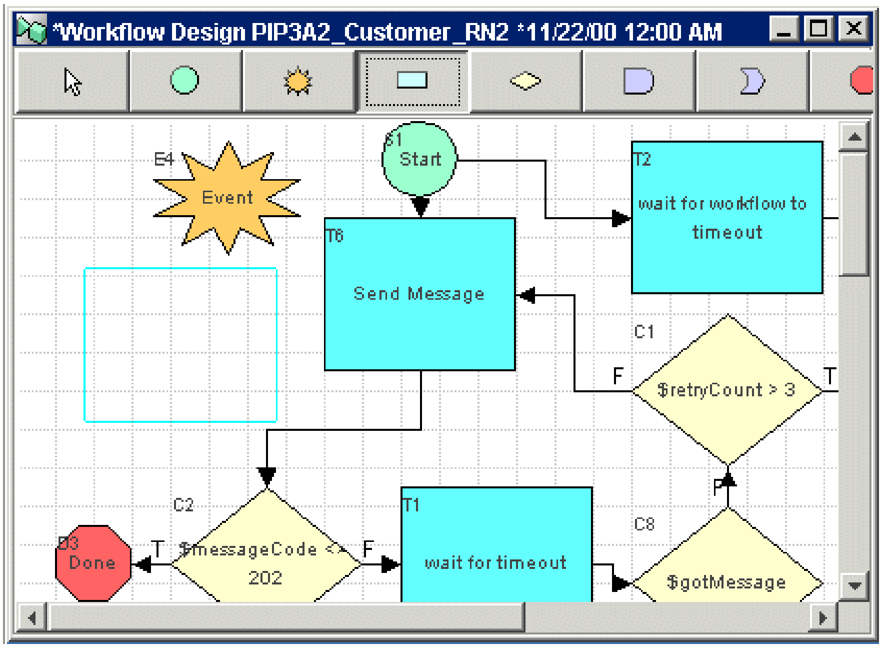
b. Create an Event node. In the toolbar, click Create Event and then click a location in the workflow diagram near the Send Business Action task.

Figure 2-4 Create an Event Node



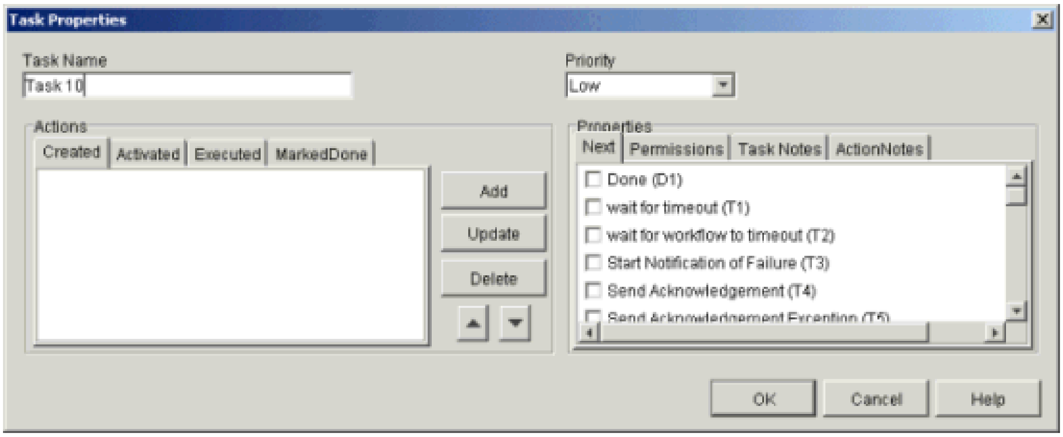
c. Create a Task node. In the toolbar, click Create Task and then click a location in the workflow diagram near the Send Business Action task.

Figure 2-5 New Task Node



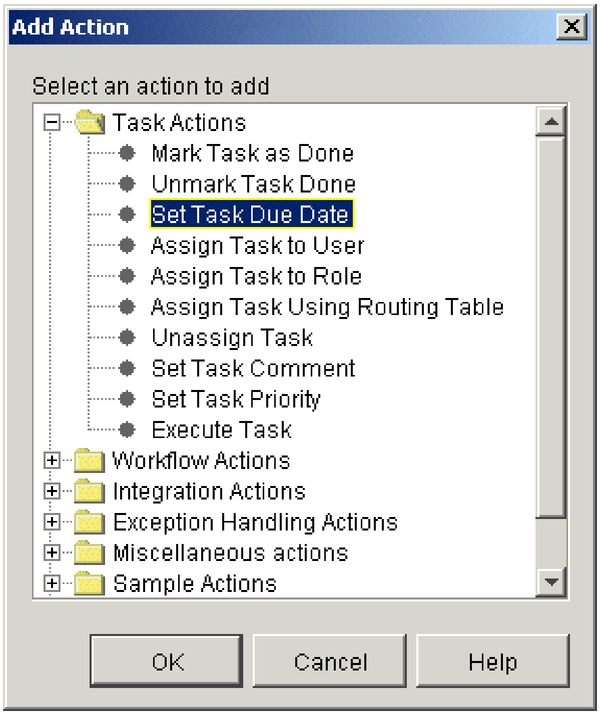
d. Double-click the Task node that you just created. The Task Properties window is displayed.

Figure 2-6 Task Properties Window



e. Enter a descriptive task name in the Task Name field, such as Wait for Http Status. Select the Created tab in the Actions section of the window, and click Add. The Add Action window is displayed.

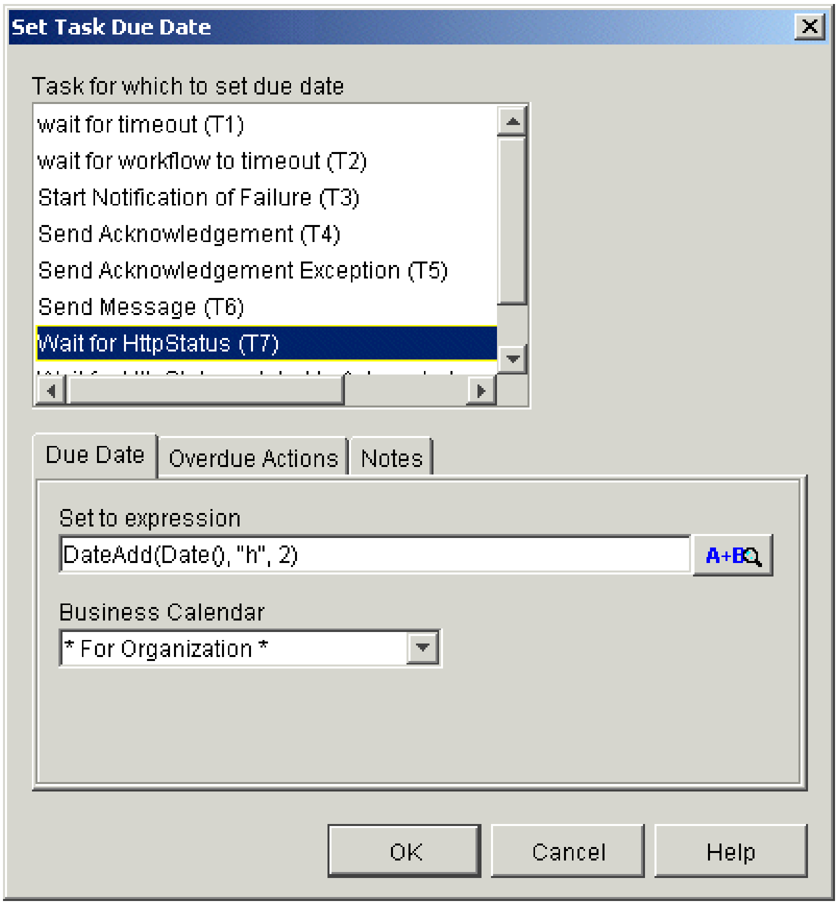
Figure 2-7 Add Action Window



f. Expand the folder named Task Actions and select Set Task Due Date.

The Set Task Due Date window is displayed.

Figure 2-8 Setting the Timeout Value



Set Task Due Date

Task for which to set due date

- wait for timeout (T1)
- wait for workflow to timeout (T2)
- Start Notification of Failure (T3)
- Send Acknowledgement (T4)
- Send Acknowledgement Exception (T5)
- Send Message (T6)
- Wait for HttpStatus (T7)

Set to expression: DateAdd(Date(), "h", 2)

Business Calendar: * For Organization *

g. From the list shown under the heading Task for which to set due date list, select the current task. (If you are continuing work with the example introduced in step 6-e, the name of the task is Wait for Http Status.)

In the Set to expression field, enter a formula that expresses the amount of time you want to designate as the timeout interval for receiving the reply. For example, to set the timeout to two hours, enter the expression shown in Figure 2-8.

Notice the second argument in the `DateAdd()` method `"h"`: it specifies the unit of time (hours). The third argument specifies the amount of time. Thus, in the example shown in Figure 2-8, the timeout interval specified is two hours.

h. Select the Overdue Actions tab and click Add.

   The Add Action window is displayed.

i. Expand the Task Actions folder, select the Mark Task Done, and click OK.

   The Mark Task as Done window is displayed.

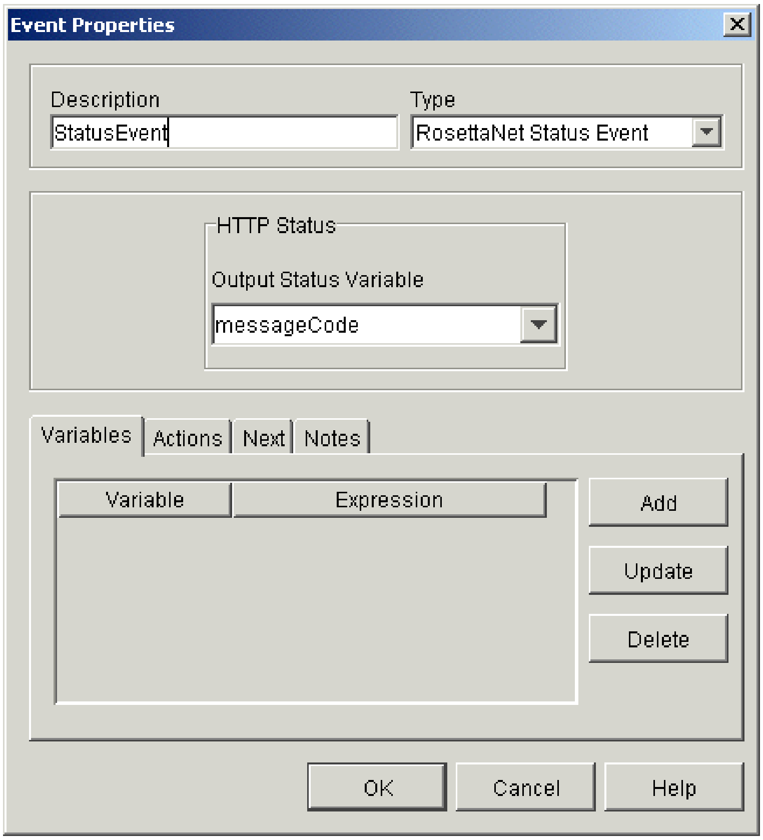
j. In the Task to mark as done list, select the current task. (If you are continuing to work with the example introduced in step 6-e, the name of the task is Wait for Http Status.) Click OK.

k. Click OK in the Task Properties window. You are returned to the Workflow Design window for you application's workflow template.

l. In the Workflow Design window, double-click the event node created in step 6-b. The Event Properties window is displayed.

m. In the Description field enter an appropriate name for the event, such as StatusEvent. Select the RosettaNet Status Event in the Type field. The Event Properties window is updated to reflect the specific fields for the RosettaNet Status Event, as shown in Figure 2-9.

Figure 2-9 Event Properties Window with RosettaNet Status Event


Event Properties

Description: StatusEvent

Type: RosettaNet Status Event

HTTP Status

Output Status Variable: messageCode

n. In the Output Status Variable field, select a variable in which to store the HTTP Status.

The Send Business Message sends a message. The RosettaNet Status Event waits until it receives the HTTP status reply, and then it stores that HTTP Status in the HTTP Status Output Status Variable. In the PIP3A2_Customer_RN2 example shown in Figure 2-5, the HTTP status reply is stored in the `messageCode` variable by the StatusEvent. Then the decision node tests the variable `messageCode` to determine whether the HTTP status is equal to 202. An HTTP status of 202 means the request has been accepted for processing, but the processing is not complete.

o. In the Event Properties window select the Actions tab and click Add. The Add Action window is displayed.

p. Expand the folder named Task Actions, select Mark Task As Done, and click OK. The Mark Task as Done window is displayed.

q. Select the Task node created in step 6-c and click OK. (If you are continuing to work with the example provided in step 6-e, then the name of the task is Wait for Http Status.)

r. In Event Properties window, click OK.

s. Remove the link from the existing task that sends the business message to the decision node. (In this example, the decision node is labeled `$messageCode <> 202.`)

   In the toolbar, click Select Shape.

   Select the link and press the delete key.

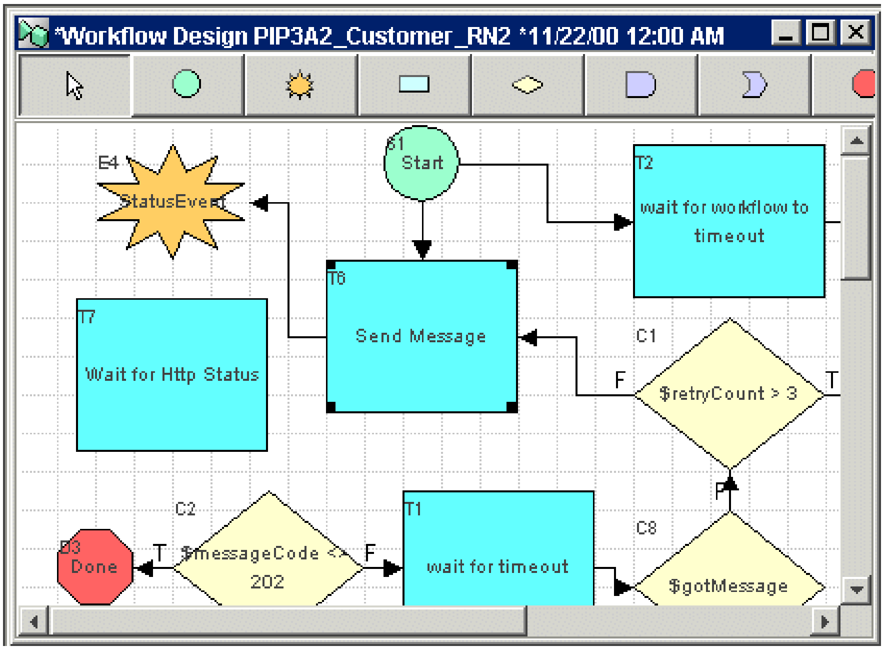
t. Link the existing task that sends the business message (the Send Message task in Figure 2-5) to the new event created in step 6-b.

   In the toolbar, click Draw Connection. Move the cursor into the middle of the existing Send Message task.

   Click the Send Message task, drag to the event created in the step 6-b, and release the mouse button. (If you are continuing to work with the example introduced in step 6-e, the name of the event is StatusEvent.)

   An arrow (pointing from the task that sends the business message to the new event) is added to the workflow design, as shown in the following figure.

Figure 2-10 Linking the Task that Sends the Business Message to the Status Event



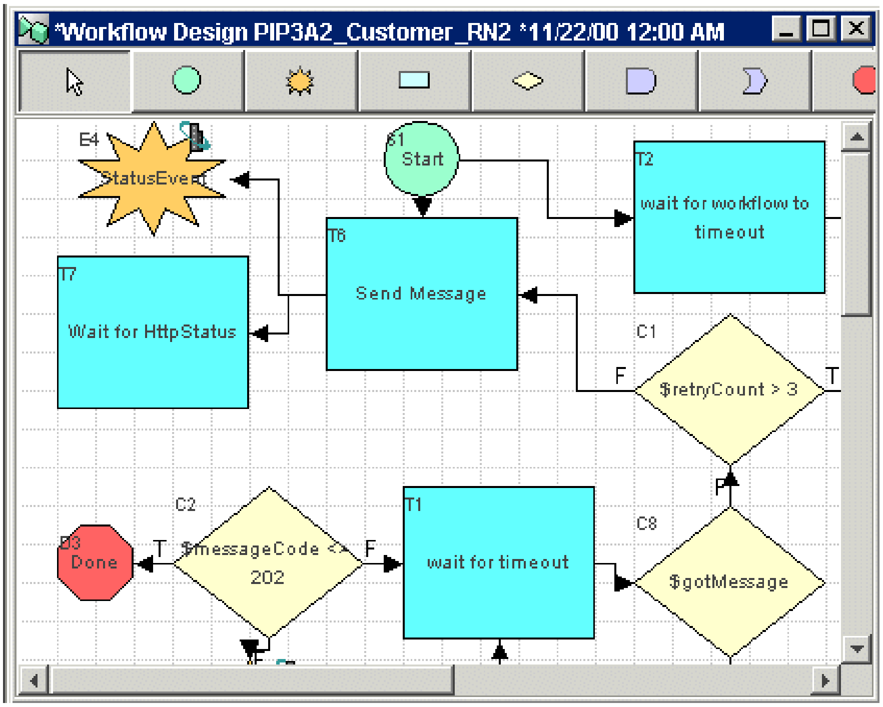
u. Link the Send Message task to the task node created in step 6-c.

In the toolbar, click Draw Connection. Move the cursor into the middle of the existing Send Message task.

Click the Send Message task, drag to the task created in the step 6-c, and release the mouse button. (If you are continuing to work with the example introduced in step 6-e, the name of the task is Wait for Http Status.)

An arrow (pointing from the Send Message task to the new task) is added to the workflow design, as shown in the following figure.

Figure 2-11 Linking the Event Node to the New Task Node



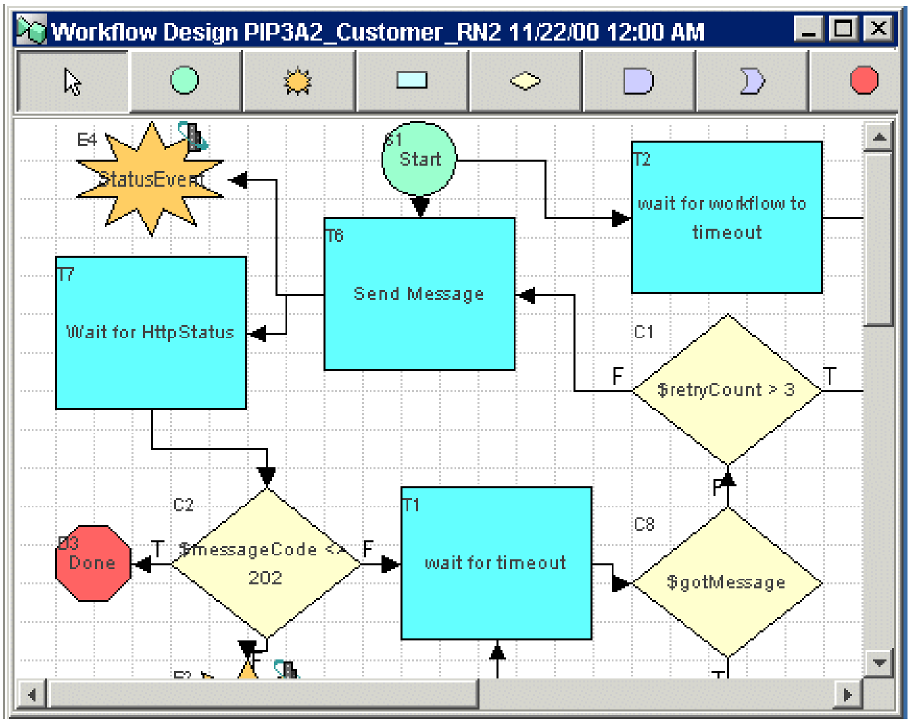
v. Link the new task node to the node that followed the original send message task. (In the example shown in Figure 2-11, link the new Wait for Http Status task to the existing decision node labeled `$messageCode <> 202`.)

In the toolbar, click Draw Connection. Move the cursor into the middle of the new task node.

Click the new task node, drag to the decision node, and release the mouse button. (In this example, the decision node is labeled `$messageCode <> 202.`)

An arrow, pointing from the new task to the exiting decision node, is added to the workflow design, as shown in the following figure.

Figure 2-12 Linking the New Task with the Existing Decision Node



You have now finished converting an instance of the Send Business Message task in a RosettaNet workflow to be compatible with WebLogic Integration 7.0.

7. Repeat step 6 a-v, for every instance of the Send Business Message task in your RosettaNet application workflows.

8. Save the Workflow by clicking the X in the top right corner of the Workflow Design window. The Workflow Changed dialog box is displayed with the question “Do you want to save changes now?” Click Yes.

9. When you have completed steps 5 to 8 for all the workflows in your application JAR file, we recommend you backup the workflows by exporting a new version of the JAR file. To do so, completing the following steps:

   a. From the Studio menu bar, choose Tools→Export Package... .

b. In the Export:Select File window, enter a full pathname to create an application JAR file that contains the converted RosettaNet workflow templates. Click Next. In the left pane, select the components to be exported. Click Export, then Close.

Note: For more detailed instructions on exporting workflows, see "Importing and Exporting Workflow Packages" in *Using the WebLogic Integration Studio*.

You have now finished converting your RosettaNet workflows.

Step 11. Deploy Your Application Views

If your application uses the application integration feature of WebLogic Integration, you must deploy your application views by following the auto-deploy procedure or by using the WebLogic Integration Application View Console. If your application does not use application integration, however, you can skip this step.

The procedure presented in this section is based on the assumption that the application views for your application are stored in the database that was converted to WebLogic Integration 7.0 in "Step 6. Migrate Your Database" on page 2-11. The procedures does not create new application views; it simply verifies that your application view settings are correct for the new environment and then deploys existing application views.

Warning: Before deploying your application views, you must configure your application integration adapter EAR file for you new domain (which you created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6). For instructions, see "Configuring Application Integration Adapter EAR Files" on page 3-3.

To deploy your application view, complete one of the following procedures:

- Deploy Using the Application View Console—Complete this procedure if you need to update your Event Router URL or if you are deploying your application view to a cluster environment.
- Auto-Deploy Application Views

Deploy Using the Application View Console

To deploy your application views using the Application View Console:

1. Go to the WebLogic Integration home directory and set the top-level WebLogic Integration environment variables by running the `setenv` script. The commands you must run to perform these tasks depend on which platform you are using:

   - Windows:

     ```
     cd c:\bea\weblogic700\integration
     setEnv.cmd
     ```

   - UNIX:

     ```
     cd /home/joe/bea/weblogic700/integration
     . setenv.sh
     ```

2. Start an instance of WebLogic Server using the new WebLogic Integration 7.0 domain (which you created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6) by completing the procedure appropriate for your platform:

- On a Windows system, you have a choice of two methods:

  - To start the server from a menu:

    Choose Start→BEA WebLogic Platform 7.0→User Projects→*domain*→Start Server.

  - To start the server from the command line:

    ```
    cd DOMAIN_HOME
    startWeblogic.cmd
    ```

- UNIX:

  ```
  cd DOMAIN_HOME
  startWebLogic
  ```

  In the `cd` command line, *DOMAIN_HOME* is the pathname of the new domain that you created in "Step 5. Create a New Domain and Configure Database Information."

3. Log on to the Application View Console:

   a. Open a new browser window.

b. Enter the URL for your system's Application View Console. The actual URL you enter depends on your system. It should conform to the following format:

   `http://host:port/wlai`

   The Application View Console Logon page is displayed.

c. Enter your WebLogic Server username and password, then click Login.

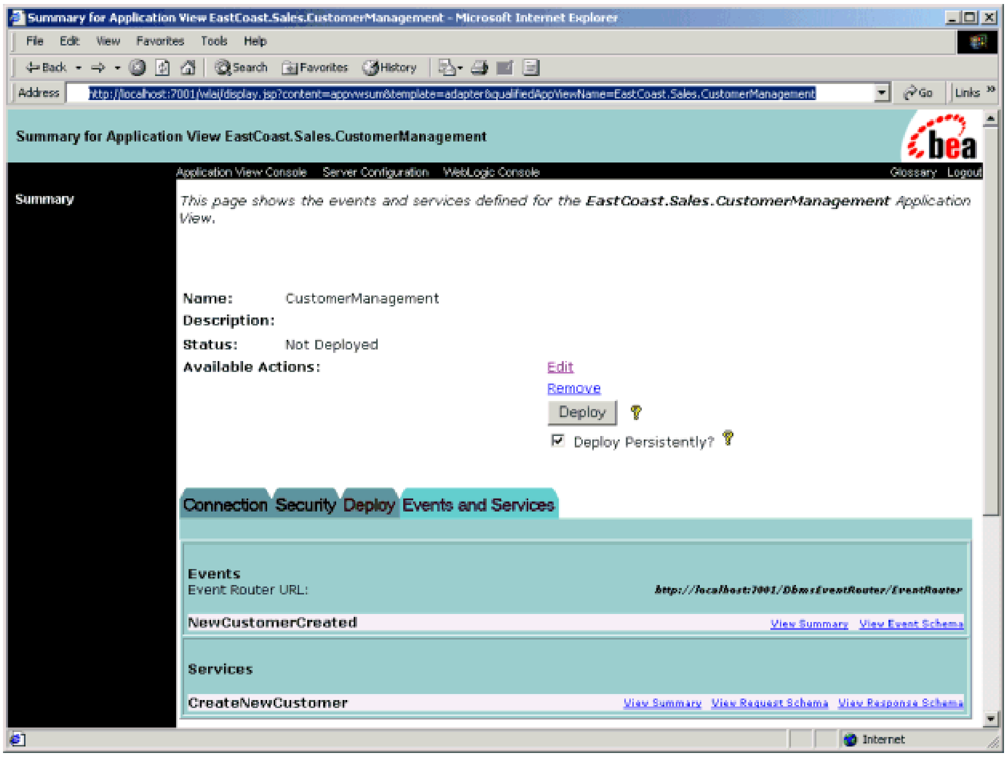
   The Application View Console is displayed. A list of application views is displayed in the Console.

Note: For more detailed instructions, see "Step 1: Log On to the Application View Console" in "Defining an Application View" in *Using Application Integration*.

4. Double-click an application view from the list. Click until you get to the end node of the application view. If the application view is currently deployed click Undeploy.

   The Summary page for the selected application view is displayed. The following figure shows the Summary page for the EastCoast.Sales.CustomerManagement application view, which is based on the example described in "Configuring Application Integration Adapter EAR Files" on page 3-3.

Figure 2-13 EastCoast.Sales.CustomerManagement Application View



Summary for Application View EastCoast.Sales.CustomerManagement - Microsoft Internet Explorer

Address: http://localhost:7001/wlai/display.jsp?content=appvwsum&template=adapter&qualifiedAppViewName=EastCoast.Sales.CustomerManagement

Summary for Application View EastCoast.Sales.CustomerManagement

This page shows the events and services defined for the ***EastCoast.Sales.CustomerManagement*** *Application View.*

Name: CustomerManagement
Description:
Status: Not Deployed
Available Actions: Edit, Remove, Deploy
☑ Deploy Persistently?

Events
Event Router URL: *http://localhost:7001/DbmsEventRouter/EventRouter*

NewCustomerCreated — View Summary, View Event Schema

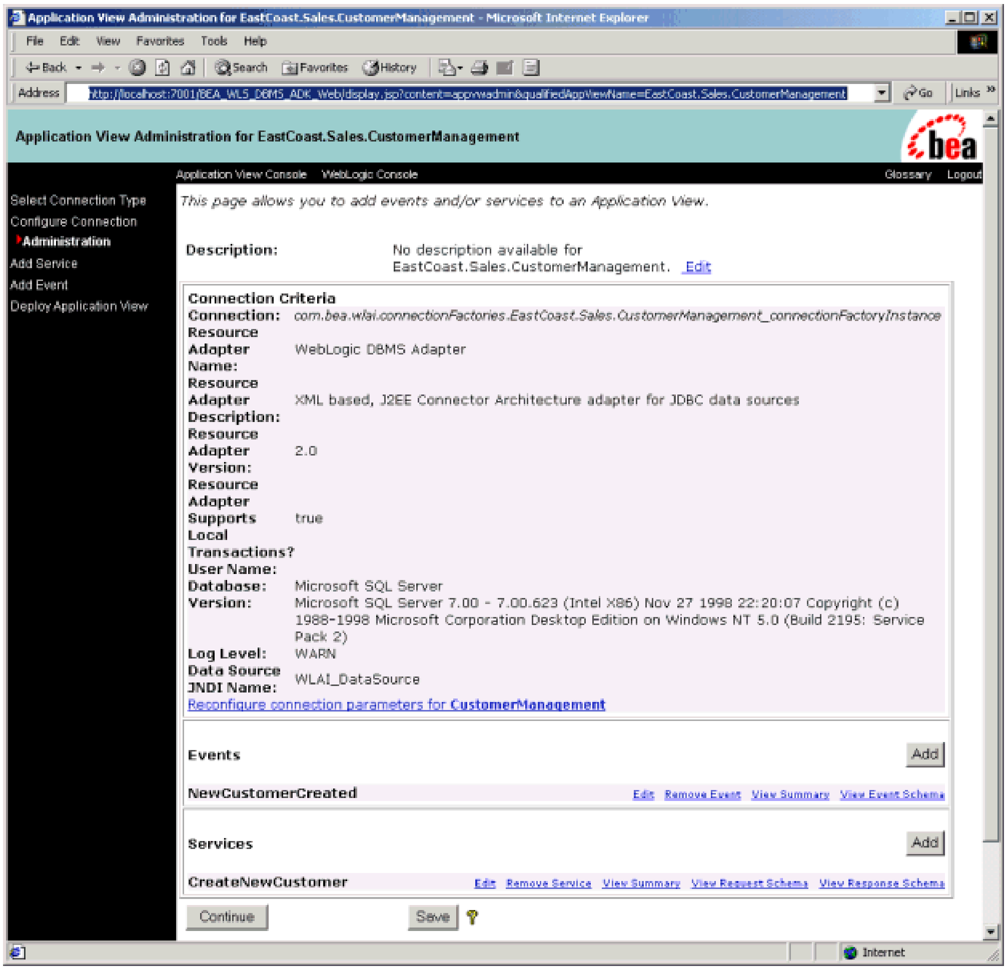
Services

CreateNewCustomer — View Summary, View Request Schema, View Response Schema

5. Click Edit.

   The Application View Administration page is displayed.

Figure 2-14 Application View Administration page



Application View Administration for EastCoast.Sales.CustomerManagement - Microsoft Internet Explorer

Address http://localhost:7001/BEA_WLS_DBMS_ADK_Web/display.jsp?content=appvwadmin&qualifiedAppViewName=EastCoast.Sales.CustomerManagement

Application View Administration for EastCoast.Sales.CustomerManagement

This page allows you to add events and/or services to an Application View.

Description: No description available for EastCoast.Sales.CustomerManagement. Edit

Connection Criteria

Connection:	com.bea.wlai.connectionFactories.EastCoast.Sales.CustomerManagement_connectionFactoryInstance
Resource Adapter Name:	WebLogic DBMS Adapter
Resource Adapter Description:	XML based, J2EE Connector Architecture adapter for JDBC data sources
Resource Adapter Version:	2.0
Resource Adapter Supports Local Transactions?	true
User Name:	
Database:	Microsoft SQL Server
Version:	Microsoft SQL Server 7.00 - 7.00.623 (Intel X86) Nov 27 1998 22:20:07 Copyright (c) 1988-1998 Microsoft Corporation Desktop Edition on Windows NT 5.0 (Build 2195: Service Pack 2)
Log Level:	WARN
Data Source JNDI Name:	WLAI_DataSource

Reconfigure connection parameters for CustomerManagement

Events

NewCustomerCreated

Services

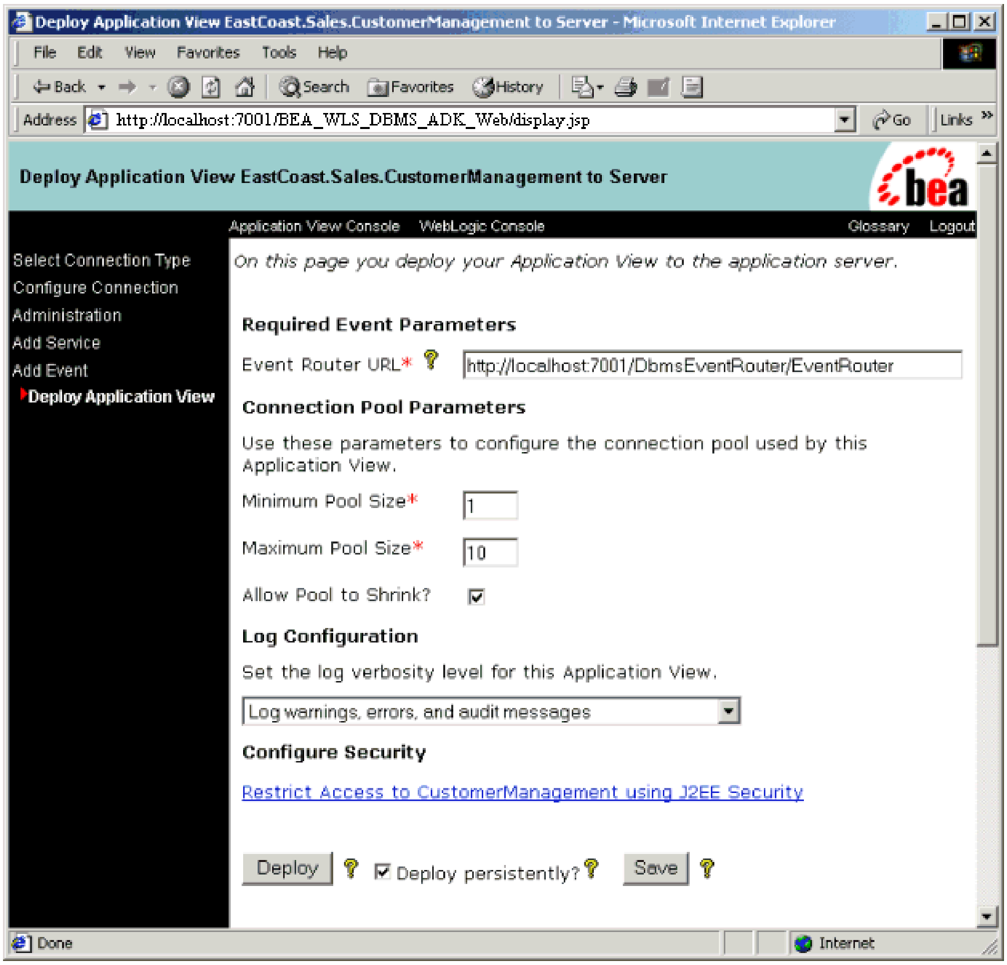
CreateNewCustomer

File Edit View Favorites Tools Help
Back Search Favorites History
Go Links
Application View Console WebLogic Console Glossary Logout
Select Connection Type
Configure Connection
Administration
Add Service
Add Event
Deploy Application View
Add
Edit Remove Event View Summary View Event Schema
Add
Edit Remove Service View Summary View Request Schema View Response Schema
Continue
Save
Internet

6. In the right pane, click Continue.

   The Deploy Application View page is displayed.

Figure 2-15 Deploy Application View


Deploy Application View EastCoast.Sales.CustomerManagement to Server - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://localhost:7001/BEA_WLS_DBMS_ADK_Web/display.jsp
Go Links
bea
Application View Console WebLogic Console Glossary Logout
Select Connection Type
Configure Connection
Administration
Add Service
Add Event
Deploy Application View
Deploy
Save
Done
Internet

Deploy Application View EastCoast.Sales.CustomerManagement to Server

On this page you deploy your Application View to the application server.

Required Event Parameters

Event Router URL* http://localhost:7001/DbmsEventRouter/EventRouter

Connection Pool Parameters

Use these parameters to configure the connection pool used by this Application View.

Minimum Pool Size* 1

Maximum Pool Size* 10

Allow Pool to Shrink? ✓

Log Configuration

Set the log verbosity level for this Application View.

Log warnings, errors, and audit messages

Configure Security

Restrict Access to CustomerManagement using J2EE Security

✓ Deploy persistently?

7. Check that the Event Router URL is correct for your WebLogic Integration 7.0 environment. Make any necessary changes in the Event Router URL field.

8. Click Deploy.

Auto-Deploy Application Views

This procedure automatically deploys application views that where stored in the database that was converted to the WebLogic Integration 7.0 in "Step 6. Migrate Your Database" on page 2-11.

To deploy your application views:

1. Backup your WebLogic Integration 7.0 `wlai.properties` file for the new domain (created in "Step 5. Create a New Domain and Configure Database Information.") by completing the following steps:

   a. Go to the *`DOMAIN_HOME`*`\wlai` directory (*`DOMAIN_HOME`* represents the pathname of the domain you created in "Step 5. Create a New Domain and Configure Database Information").

   b. Rename the `wlai.properties` file to `wlai.properties.70` by completing the procedure appropriate for your platform:

   - Windows:

     ```
     rename wlai.properties wlai.properties.70
     ```

   - UNIX:

     ```
     mv wlai.properties wlai.properties.70
     ```

2. Copy the WebLogic Integration 2.1 `wlai.properties` file into the new domain (created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6). For example, if the WebLogic Integration 2.1 EAI Domain was used in your application, copy the `wlai.properties` file from the *`WLI_HOME`*`\config\eaidomain\wlai` directory to the *`DOMAIN_HOME`*`\wlai` directory, where *`WLI_HOME`* represents the WebLogic Integration 2.1 home directory, and *`DOMAIN_HOME`* represents the pathname of the domain you created in "Step 5. Create a New Domain and Configure Database Information".

3. In a text editor, open the `wlai.properties` file. As the value of the `wlai.appView.deploy` property, specify a comma-delimited list of the application views you want to deploy. For example, to deploy the `EastCoast.Sales.CustomerManagement` and `WestCoast.Sales.CustomerManagement` applications views, set the `wlai.appView.deploy` property as shown in the following code listing.

   ```
   wlai.appView.deploy=EastCoast.Sales.CustomerManagement,
   WestCoast.Sales.CustomerManagement
   ```

4. Remove all the other properties except the `wlai.appView.deploy` property from the `wlai.properties` file.

5. Start the instance of WebLogic Server using the new WebLogic Integration 7.0 domain created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6 by completing the procedure appropriate for your platform.

- On a Windows system, you have a choice of two methods:
  - To start the server from a menu:

    Choose Start→BEA WebLogic Platform 7.0→User Projects→*`domain`*→Start Server.
  - To start the server from the command line:

    ```
    startWeblogic.cmd
    ```

  When the instance of WebLogic Server is booted, the listed application views will automatically be deployed.

6. Shut down the WebLogic Server instance for the new WebLogic Integration 7.0 domain (created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6), by completing the procedure appropriate for your platform:
   - Windows:

     ```
     cd DOMAIN_HOME
     stopWeblogic.cmd
     ```
   - UNIX:

     ```
     cd DOMAIN_HOME
     stopWebLogic
     ```

   In both `cd` command lines, *`DOMAIN_HOME`* represents the pathname of the new domain (created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6), and *`domain`* represents the directory you created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6.

7. Restore the WebLogic Integration 7.0 `wlai.properties` file for the new domain by completing the following steps:

a. Go to the `DOMAIN_HOME\wlai` directory (*`DOMAIN_HOME`* represents the pathname of the domain you created in “Step 5. Create a New Domain and Configure Database Information”).

b. Rename the `wlai.properties` file to `wlai.properties.21` by completing the procedure appropriate for your platform:

   Windows:

   ```
   rename wlai.properties wlai.properties.21
   ```

   UNIX:

   ```
   mv wlai.properties wlai.properties.21
   ```

c. Copy the `wlai.properties.70` file to `wlai.properties` by completing the procedure appropriate for your platform:

   Windows:

   ```
   rename wlai.properties.70 wlai.properties
   ```

   UNIX:

   ```
   mv wlai.properties.70 wlai.properties
   ```

Step 12. Start and Test Your WebLogic Integration Application

Start, run, and test your application using WebLogic Integration 7.0.

If you get an exception similar to the following, when starting the WebLogic Integration 7.0 server, you must regenerate your `SerializedSystemIni.dat` file.

```
weblogic.security.internal.FileUtilsException: Couldn't rename
DOMAIN_HOME\SerializedSystemIni.dat to
DOMAIN_HOME\SerializedSystemIni.dat42698.old
```

In this messages *`DOMAIN_HOME`* represents the pathname of the new domain that you created in “Step 5. Create a New Domain and Configure Database Information” on page 2-6."

For the procedure for generating a new `SerializedSystemIni.dat` file, see "Generating a New SerializedSystemIni.dat File" on page 3-8.

Importing WebLogic Integration 2.1 Repository Data Files

In "Step 6. Migrate Your Database" on page 2-11, the WebLogic Integration 2.1 repository data stored in your database was converted to the WebLogic Integration 7.0 format. This section is required only if you want to import additional repository information stored in repository files for WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1.

As part of the database migration in "Step 6. Migrate Your Database" on page 2-11, a new system identity, `wlisystem`, was created with a password of `wlisystem`. The new identity was stored in both the WebLogic Integration repository and the security realm. The `wlisystem` system identity replaces the `wlcsystem` and `wlpisystem` system identities for WebLogic Integration 2.1 and WebLogic Integration 2.1 SP1.

Before importing a WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 repository data file into the WebLogic Integration 7.0 repository, you must change the `system-password` attribute of the `LC` element in the repository data file to reflect the current password for `wlisystem`. (In WebLogic Integration 2.1 and WebLogic Integration 2.1 SP1, the value specified by the `system-password` attribute of the `LC` element is the password for the `wlcsystem` system identity.)

Set the `system-password` attribute of the `WLC` element in the repository data file to reflect the password of the `wlisystem` system identity, as shown in the following listing. (The password for the `wlisystem` system identity is `wlisystem`, unless you changed the `wlisystem` password using the B2B Console.)

Listing 2-6 Set system-password

```
<?xml version="1.0"?>
<!DOCTYPE wlc SYSTEM "WLC.dtd">
<wlc
       large-msg-support-on="OFF"
       large-msg-min-size="0"
       large-msg-location="c:\temp"
```

```
        system-password="wlisystem"
        >
```

The same password must be specified for `wlisystem` in both the WebLogic Integration repository and the security realm. Failure to keep the two copies synchronized can cause problems when you boot the WebLogic Integration instance for WebLogic Integration. To make sure that the two copies remain synchronized, use the B2B Console whenever you need to change the password. When you make a change, the B2B Console automatically saves the password in both the security realm and the WebLogic Integration repository.

When you import a repository data file, the value of the `system-password` attribute overrides the password for the `wlisystem` system identity in the repository; the password for the `wlisystem` system identity in the security realm is not updated. If you change the `wlisystem` password using the WebLogic Integration Administration Console, the password for `wlisystem` is changed only in the security realm.

CHAPTER

3 Other Migration Topics

This section provides information about the following topics:

- Clustering
- Configuring Application Integration Adapter EAR Files
- Trusted Relationships
- RosettaNet Schema Changes

Clustering

Clustering is supported by the business process management (BPM) and application integration functionality provided, in turn, by WebLogic Integration 2.1, 2.1 SP1, and 7.0. The configuration requirements for a 7.0 cluster application, however, are different from those for a 2.1 or 2.1 SP1 cluster application, resulting in differences in the clustering sections of the `config.xml` files for the 2.1 and 7.0 releases. In addition, WebLogic Integration 7.0 supports clustering of the business-to-business functionality provided by WebLogic Integration.

If you are migrating a WebLogic Integration cluster application from Release 2.1 or 2.1 SP1 to Release 7.0, you must first create a new WebLogic Integration domain with clustering enabled, and then migrate application-specific entities (such as JMS queues) to it, as described in "Step 7. Migrate Components of Your WebLogic Integration Application" on page 2-14. For more information about creating a WebLogic Integration domain with clustering enabled, see "Configuring a Clustered Deployment" in *Deploying BEA WebLogic Integration Solutions*.

The configuration requirements related to clustering have been changed in Release 7.0, for the following BPM resources:

- JMS connection factories
- JMS servers
- JMS destinations, which are now distributed destinations for WebLogic Integration 7.0. The following distributed destinations are required for a BPM-cluster configuration:
  - BPM audit topic (`com.bea.wli.bpm.AuditTopic`)
  - BPM error topic (`com.bea.wli.bpm.ErrorTopic`)
  - BPM notifications topic (`com.bea.wli.bpm.NotifyTopic`)
  - BPM event queue (`com.bea.wli.bpm.EventQueue`)
  - BPM validating event queues (`com.bea.wli.bpm.ValidatingEventQueue`)
  - wli error queue (`com.bea.wli.FailedEventQueue`)

The configuration requirements related to clustering have been changed, in Release 7.0, for the following application integration resources:

- JMS connection factories
- JMS servers
- JMS destinations, which are now distributed destinations for WebLogic Integration 7.0. The following distributed destinations are required for an application integration-cluster configuration:
  - wlai event queue (`com.bea.wlai.EVENT_QUEUE`)
  - wlai event topic (`com.bea.wlai.EVENT_TOPIC`)
  - wlai asynchronous request queue (`com.bea.wlai.ASYNC_REQUEST_QUEUE`)
  - wlai asynchronous response queue (`com.bea.wlai.ASYNC_RESPONSE_QUEUE`)
  - wli error queue (`com.bea.wli.FailedEventQueue`)

For more information, see “Understanding JMS Resources" in “Understanding WebLogic Integration Clusters” in *Deploying BEA WebLogic Integration Solutions*.

Configuring Application Integration Adapter EAR Files

This section provides a procedure for configuring an EAR file for an application integration adapter to be used in a sample WebLogic Integration 7.0 application. This sample is based on the following scenario.

A WebLogic Integration 2.1 customer called Acme is migrating a WebLogic Integration application to WebLogic Integration 7.0. The Acme application contains one application view called EastCoast.Sales.CustomerManagement, which uses a relational database adapter called ACME_DBMS. The ACME_DBMS adapter was developed by Acme's IT developers.

To configure an application integration adapter EAR file for the Acme application, do one of the following:

- Edit the Configuration File for the New Domain
- Invoke the WebLogic Server Deployer
- If you are deploying your adapter EAR file in a cluster environment, you must follow the instructions in "Deploying Adapters" in "Understanding WebLogic Integration Clusters" in *Deploying BEA WebLogic Integration Solutions*.

Edit the Configuration File for the New Domain

Open the `config.xml` file for the new domain (the one created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6) and add the configuration information for the adapter. To do so, complete the following procedure:

1. In a text editor, open the WebLogic Integration 2.1 application `config.xml` file that deployed the adapter EAR file.
2. Copy the section of the file that deploys your adapter EAR file (part of the Application element) as shown in the following listing for the Acme example.

Listing 3-1 WebLogic Integration 2.1 Deployment of an Adapter Ear File

```
<Application Deployed="true" Name="ACME_DBMS" Path="d:\ACME_DBMS.ear">
   <ConnectorComponent Name="ACME_DBMS"
      Targets="myserver"
      URI="ACME_DBMS.rar"/>
   <WebAppComponent Name="DbmsEventRouter"
      Targets="myserver"
      URI="DbmsEventRouter.war"/>
   <WebAppComponent Name="ACME_DBMS_Web"
      Targets="myserver"
      URI="ACME_DBMS_Web.war"/>
</Application>
```

3. Open the `config.xml` file for the new WebLogic Integration 7.0 domain (the one you created in "Step 5. Create a New Domain and Configure Database Information" on page 2-6) and paste the Application element into it.
4. Add the `TwoPhase` attribute to the Application element, as shown, in bold, in the following figure. Set the value of this element to `"true"`.

Listing 3-2 Add the TwoPhase Attribute to the Application Element

```
<Application Deployed="true" Name="ACME_DBMS" TwoPhase="true"
Path="d:\ACME_DBMS.ear">
```

5. If the location of the EAR file has changed, update the `Path` attribute of the `Application` element for your adapter as shown, in bold, in the following figure.

Listing 3-3 Update the Path of the EAR file

```
<Application Deployed="true" Name="ACME_DBMS" TwoPhase="true"
Path="c:\adk\ACME_DBMS.ear">
```

Warning: If your adapter EAR file is located in the home directory of your WebLogic Integration 2.1 or WebLogic Integration 2.1 SP1 installation (for example, `%WLI_HOME%\lib` on a Windows system), we recommend you move it to a location outside the WebLogic Integration 2.1 installation, such as `c:\adk`.

6. Update the setting of the `Targets` attribute of the `Application` element for your adapter so it matches the name of the server for the new domain (the one you created in “Step 5. Create a New Domain and Configure Database Information” on page 2-6). For example, if your domain was created from the WLI domain template, the name of the server is set to `myserver`, as shown, in bold, in the following example listing.

Listing 3-4 Get the Server Name from the Config.xml File

```
<Domain Name="mydomain">
   <!-- Domain log -->
   <Log FileName="c:/bea/user_projects/mydomain/logs/mydomain.log"
Name="mydomain"/>
   <Server ListenPort="7001"
      Name="myserver"
      TransactionLogFilePrefix="c:/bea/user_projects/mydomain/logs/"
      NativeIOEnabled="true">
.
.
.
</Domain>
```

Update the setting of the `Targets` attribute of the `Application` element for your adapter, as shown in bold in the following listing.

Listing 3-5 Update the Targets Attribute

```
<Application Deployed="true" TwoPhase="true" Name="ACME_DBMS"
Path="c:\adk\ACME_DBMS.ear">
   <ConnectorComponent Name="ACME_DBMS"
      Targets="myserver"
      URI="ACME_DBMS.rar"/>
   <WebAppComponent Name="DbmsEventRouter"
      Targets="myserver"
```

```
        URI="DbmsEventRouter.war"/>
    <WebAppComponent Name="ACME_DBMS_Web"
        Targets="myserver"
        URI="ACME_DBMS_Web.war"/>
</Application>
```

Invoke the WebLogic Server Deployer

Use the WebLogic Server deployer command-line utility to deploy the adapter EAR. For example, to deploy the `ACME_DBMS.ear` file on a Windows system, enter the following command:

```
java -classpath %BEA_HOME%\weblogic700\lib\weblogic.jar
weblogic.Deployer -adminurl
http://wls_admin_host:wls_admin_port -user wls_admin_user
-password wls_admin_pass -verbose -stage -activate -source
c:\adk\ACME_DBMS.ear -name ACME_DBMS -targets server_name
```

Table 3-1 Substitutable Strings Used in Deployer Example

This string . . .	Represents . . .
BEA_HOME	Directory in which BEA product software is installed, such as `c:\bea` on a Windows system
wls_admin_host	Hostname or IP address of the WebLogic Server instance being used
wls_admin_port	Port number of the WebLogic Server instance being used
wls_admin_user	Username used to log in to the WebLogic Server Adminstration Console. (This name is the username you specified when you used the Configuration Wizard to create a domain in "Step 5. Create a New Domain and Configure Database Information" on page 2-6.)
wls_admin_pass	Password for the login to the WebLogic Server Adminstration Console. (This password is the one you specified when you used the Configuration Wizard to create a domain in "Step 5. Create a New Domain and Configure Database Information" on page 2-6.)

This string . . .	Represents . . .
server_name	Name of the server for the domain you created in “Step 5. Create a New Domain and Configure Database Information” on page 2-6.

For more information about the Deployer utility, see “Deployment Tools and Procedures” in “WebLogic Server Deployment” in *Developing WebLogic Server Applications*. This document is available, in the BEA WebLogic Server document set, at the following URL:

`http://edocs.bea.com/wls/docs70/programming/deploying.html`

Warning: The DBMS Sample Adapter has been changed in Release 7.0 of WebLogic Integration. This change may affect your Application View for this adapter. For details, see “Changes to DBMS Sample Adapter for Services Not Requiring Request Data” in “Migrating Adapters to WebLogic Integration 7.0” in *Developing Adapters*.

For more information about application integration adapter migration, see “Migrating Adapters to WebLogic Integration 7.0” in *Developing Adapters*.

Trusted Relationships

The security model for WebLogic Server trusted relationships has been changed for Release 7.0 of WebLogic Server. A WebLogic Server 7.0 server (acting as a client) authenticates another WebLogic 7.0 server in another domain. Both domains must have the same Credential attribute on their `SecurityConfigurationMBean`. If they do not, the following exception is thrown: `java.lang.SecurityException Invalid Subject`.

You can change the Credential attribute of the domain by doing one of the following:

- Use the WebLogic Server Administration Console to change the credential. For details, see the *BEA WebLogic Server Administration Guide*. This document is available, in the BEA WebLogic Server document set, at the following URL:

  `http://e-docs.bea.com/wls/docs70/admin.html`

- In a text editor, open the `config.xml` file and change the credential for the following element:

  ```
  <SecurityConfiguration Credential=”foobar” Name=”mydomain”/>
  ```

  Note: Any cleartext credential is encrypted the first time the `config.xml` file is generated.

Generating a New SerializedSystemIni.dat File

If, when starting the WebLogic Integration 7.0 server, you get an exception similar to the following, you must regenerate your `SerializedSystemIni.dat` file.

```
weblogic.security.internal.FileUtilsException: Couldn't
rename DOMAIN_HOME\SerializedSystemIni.dat to
DOMAIN_HOME\SerializedSystemIni.dat42698.old
```

In these messages, *`DOMAIN_HOME`* represents the pathname of the new domain you created in “Step 5. Create a New Domain and Configure Database Information” on page 2-6.

To generate a new `SerializedSystemIni.dat` file for your domain, complete the following steps:

1. Go to the directory in which your domain resides:

   ```
   cd DOMAIN_HOME
   ```

   `DOMAIN_HOME` represents the pathname of the new domain that you created in “Step 5. Create a New Domain and Configure Database Information” on page 2-6.

2. Open the `config.xml` file and remove any encrypted sections from the Credential attributes. For example, remove the configuration section shown in Listing 3-6 and replace it with the configuration section shown in Listing 3-7.

Listing 3-6 Credential Elements with Encryption

```
<EmbeddedLDAP
    Credential="{3DES}CUnX5jDsmxluwzO45cb+kErCo+3pa92oXcPZl8L23pwx"
Name="mydomain"/>
    <SecurityConfiguration Credential="{3DES}aKaA3CEY4TIx" Name="mydomain"/>
```

Listing 3-7 Encryption Removed from Credential Elements

```
<EmbeddedLDAP
    Credential="" Name="mydomain"/>
    <SecurityConfiguration Credential="" Name="mydomain"/>
```

3. Make a copy of the `fileRealm.properties` file and call the copy `fileRealm.properties.src`, by running the command appropriate for your platform:
   - Windows:
     ```
     copy fileRealm.properties fileRealm.properties.src
     ```
   - UNIX:
     ```
     cp fileRealm.properties fileRealm.properties.src
     ```
4. Delete the `boot.properties` file by running the command appropriate for your platform:
   - Windows:
     ```
     delete boot.properties
     ```
   - UNIX:
     ```
     rm boot.properties
     ```
5. Rename the `SerializedSystemIni.dat` file by running the command appropriate for your platform:
   - Windows:
     ```
     rename SerializedSystemIni.dat SerializedSystemIni.dat.old
     ```
   - UNIX:
     ```
     mv SerializedSystemIni.dat SerializedSystemIni.dat.old
     ```

6. Open the `fileRealm.properties.src` file and change all the hashed passwords to cleartext passwords. (Passwords are set by the `user.`*`user_name`* element, where *`user_name`* represents the user identity, as shown, in bold, in the following partial listing.) For example, remove the password encrypted in Listing 3-8 and replace it with the text version, as shown in Listing 3-9.

Listing 3-8 Partial Listing of the fileRealm.properties file with Password Encrypted

```
group.ConfigureSystem=admin,joe,mary,guest,wlisystem
acl.lookup.weblogic.jndi.weblogic.fileSystem=everyone
acl.enablermonitor.WLCAdmin=admin
user.joe=0xa078cb45e6f6c4eefdd1f14495ff739b5536904c
group.DeleteTemplate=admin,joe,mary,guest,wlisystem
group.AdministerUser=admin,joe,mary,guest,wlisystem
group.Deployers=Administrators
```

Listing 3-9 Partial Listing of the fileRealm.properties file with Password in Cleartext

```
group.ConfigureSystem=admin,joe,mary,guest,wlisystem
acl.lookup.weblogic.jndi.weblogic.fileSystem=everyone
acl.enablermonitor.WLCAdmin=admin
user.joe=password
group.DeleteTemplate=admin,joe,mary,guest,wlisystem
group.AdministerUser=admin,joe,mary,guest,wlisystem
group.Deployers=Administrators
```

7. Rename the `SerializedSystemIni.dat` file by running the command appropriate for your platform:
   - Windows:

     ```
     rename SerializedSystemIni.dat SerializedSystemIni.dat.old
     ```
   - UNIX:

     ```
     mv SerializedSystemIni.dat SerializedSystemIni.dat.old
     ```

8. Use the `FileRealm` utility to regenerate the `SerializedSystemIni.dat` file. Specifically, run the following command:

   ```
   java weblogic.security.internal.acl.FileRealm
   fileRealm.properties SerializedSystemIni.dat
   ```

RosettaNet Schema Changes

WebLogic Integration provides schema validation of RosettaNet messages. The schemas provided for this validation with WebLogic Integration 2.1 and WebLogic Integration 2.1 SP1 are based on the World Wide Web Consortium (W3C) 2000 XML Schema Definitions (XSD) schema. The schemas provided for this validation with WebLogic Integration 7.0 are based on the World Wide Web Consortium (W3C) 2001 XML Schema Definitions (XSD) schema.

A RosettaNet message is made up of two parts: a message header XML document (Service-Header) and an XML business document or payload. The XSD schemas used for validating both the message header XML document and the XML business document have been converted to use the W3C 2001 XSD schema. If you have developed schemas for WebLogic Integration message validation, you must update them to use the 2001 XSD schema. (The schema must start with the schema element shown in Listing 3-11 or Listing 3-12.) WebLogic Integration 7.0 supports only the validation of RosettaNet messages against 2001 XSD schemas.

RosettaNet message validation schemas are located in the *`WLI_HOME`*`\lib\xmlschema\rosettanet` directory. (*`WLI_HOME`* represents the WebLogic Integration home directory.)

2000 XSD schemas start with the schema element shown in the following listing.

Listing 3-10   2000 XSD Schema Element

```
<schema xmlns="http://www.w3.org/2000/10/XMLSchema">
       <!-- schema content goes here -->
</schema>
```

2001 XSD schemas start with a schema element similar to the one shown in Listing 3-11 or Listing 3-12. The XSD schema element in Listing 3-12 uses the namespace `xsd` to fully qualify elements of the XSD schema.

Listing 3-11 2001 XSD Schema Element

```
<schema xmlns="http://www.w3.org/2001/XMLSchema">
       <!-- schema content goes here -->
</schema>
```

Listing 3-12 2001 XSD Schema Element with the xsd Namespace

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
       <!-- schema content goes here -->
</schema>
```

xml: Namespace

WebLogic Integration 7.0 supports the use of the `xml:` namespace. The `xml:` namespace is a standard W3C namespace that contains the following attribute names:

- `xml:lang`—identifies the natural language being used
- `xml:space`—specifies whether white space is significant to the element
- `xml:base`—defines the base URIs for parts of XML documents

In WebLogic Integration 2.1 or 2.1 SP1 XSD schema files, `xml:lang` is converted to a string, as follows:

```
<xsd:attribute name=”xml:lang” type=”xsd:string”/>
```

In WebLogic Integration 7.0 XSD schema files, the value of `xml:lang` is determined by `xml:` namespace. If you develop your own schemas for message validation, change the `xml:lang` attribute as shown in the following listing:

```
<xsd:attribute ref=”xml:lang”/>
```

By default, WebLogic Integration 7.0 obtains the `xml:` namespace XSD schema from the local filesystem, instead of from the following standard `xml:` namespace URL:

```
http://www.w3.org/2001/xml.xsd
```

Looking up the namespace on the local filesystem, allows WebLogic Integration to improve performance and reduce network traffic.

By obtaining schema from a local filesystem instead of a URL, WebLogic Integration 7.0 can reduce network traffic and thus, improve performance. For example, the `0A1_MS_V02_00_FailureNotification.xsd` schema sets the value of the `schemaLocation` attribute to `schemas/xml.xsd`, as shown in the following listing. This setting means that WebLogic Integration will get the `xml:` namespace schema from the local file system.

Listing 3-13 0A1_MS_V02_00_FailureNotification.xsd Using Local Schema

```
<?xml version = "1.0" encoding = "UTF-8"?>
<xsd:schema xmlns:xsd = "http://www.w3.org/2001/XMLSchema">
<xsd:import namespace = "http://www.w3.org/XML/1998/namespace"
schemaLocation = "schemas/xml.xsd"/>
.
.
.
```

In Listing 3-14, the value of the `schemaLocation` attribute is `http://www.w3.org/2001/xml.xsd`. When this setting is used, WebLogic Integration optains the `xml:` namespace schema using the URL.

Listing 3-14 0A1_MS_V02_00_FailureNotification.xsd Using URL Schema

```
<?xml version = "1.0" encoding = "UTF-8"?>
<xsd:schema xmlns:xsd = "http://www.w3.org/2001/XMLSchema">
<xsd:import namespace = "http://www.w3.org/XML/1998/namespace"
schemaLocation = "http://www.w3.org/2001/xml.xsd"/>
.
.
.
```

If you prefer to validate the `xml:` namespace from the URL, you can replace the XSD schema files in the *`WLI_HOME`*`\lib\xmlschema\rosettanet` directory with a set of XSD schema files that set the value of the `schemaLocation` attribute to the standard `xml:` namespace URL. To do so, complete the following steps:

1. Go to the RosettaNet XSD schema directory by running the command appropriate for your platform:
   - Windows:

     ```
     cd WLI_HOME\lib\xmlschema\rosettanet
     ```
   - UNIX:

     ```
     cd WLI_HOME/lib/xmlschema/rosettanet
     ```

   On both command lines, *`WLI_HOME`* represents the WebLogic Integration home directory.
2. Create a backup directory by running the command appropriate for your platform:
   - Windows:

     ```
     md backup
     ```
   - UNIX:

     ```
     mkdir backup
     ```
3. Move the XSD schema files that point to the local `xml:` namespace schema file to the backup directory by completing the command appropriate for your platform:
   - Windows:

     ```
     move *.xsd backup
     ```
   - UNIX:

     ```
     mv *.xsd backup
     ```
4. Extract the XSD schema files that point to the standard `xml:` namespace URL:

   ```
   jar xvf schemas.jar
   ```

Additional 2001 XSD Schema Changes

If you develop your own schemas for message validation, you may need to make the following changes to your schemas:

- Replace the `appInfo` element with the `documentation` element. This replacement was made in WebLogic Integration 7.0 schemas to accommodate a change in the DTD grammar.
- Remove the `content=”elementOnly”` and `content=”textOnly”` attributes, as shown in the following example listings.

Listing 3-15 Example 2000 XSD Schema (WebLogic Integration 2.1 or 2.1 SP1)

```
<xsd:complexType content = "elementOnly">
<xsd:choice>
       <xsd:element ref = "GlobalLocationIdentifier"/>
       <xsd:group ref = "StreetAddress"/>
       <xsd:group ref = "GlobalLocationIdAndStreetAddress"/>
</xsd:choice>
```

Listing 3-16 Example 2001 XSD Schema (WebLogic Integration 7.0)

```
<xsd:complexType>
<xsd:choice>
       <xsd:element ref = "GlobalLocationIdentifier"/>
       <xsd:group ref = "StreetAddress"/>
       <xsd:group ref = "GlobalLocationIdAndStreetAddress"/>
</xsd:choice>
```