



BEA WebLogic Process Integrator Tutorial

BEA WebLogic Process Integrator Release 1.1
Document Edition 1.1
October 2000

Copyright

Copyright © 2000 BEA Systems, Inc. All Rights Reserved.

Restricted Rights Legend

This software and documentation is subject to and made available only pursuant to the terms of the BEA Systems License Agreement and may be used or copied only in accordance with the terms of that agreement. It is against the law to copy the software except as specifically allowed in the agreement. This document may not, in whole or in part, be copied photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form without prior consent, in writing, from BEA Systems, Inc.

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the BEA Systems License Agreement and in subparagraph (c)(1) of the Commercial Computer Software-Restricted Rights Clause at FAR 52.227-19; subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, subparagraph (d) of the Commercial Computer Software--Licensing clause at NASA FAR supplement 16-52.227-86; or their equivalent.

Information in this document is subject to change without notice and does not represent a commitment on the part of BEA Systems. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. FURTHER, BEA Systems DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE SOFTWARE OR WRITTEN MATERIAL IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE.

Trademarks or Service Marks

BEA, WebLogic, Tuxedo, and Jolt are registered trademarks of BEA Systems, Inc. The E-Commerce Transactions Company, BEA E-Commerce Transaction Platform, BEA Builder, BEA Manager, BEA eLink, BEA WebLogic Commerce Server, BEA WebLogic Personalization Server, BEA WebLogic Process Integrator, BEA WebLogic Collaborate, BEA WebLogic Enterprise, and BEA WebLogic Server are trademarks of BEA Systems, Inc.

All other product names may be trademarks of the respective companies with which they are associated.

BEA WebLogic Process Integrator Tutorial

Document Edition	Part Number	Date	Software Version
1.1	N/A	October 2000	1.1

Contents

About This Document

What You Need to Know	vi
e-docs Web Site	vi
How to Print the Document	vi
Related Information	vii
Contact Us!	vii
Documentation Conventions	ix

1. Overview of the Example

Description	1-2
About the Example	1-2
Main Steps	1-5

2. Defining the Order Processing Workflow

Overview of the Order Processing Workflow	2-2
Creating a New Workflow Template	2-5
Creating a New Workflow Template Definition	2-9
Drawing the Flow	2-11
Placing Node Icons (Shapes) in the Drawing Area	2-12
Connecting the Nodes	2-14
Renaming Nodes	2-16
Task 1 (T1) Node	2-16
Task 2 (T2) Node	2-17
Event 1 (E1) Node	2-18
Event 2 (E2) Node	2-19
Defining Variables	2-21
Defining Business Operations	2-26
Defining Tasks	2-32
Defining the Confirm Order Task	2-32
Setting Up the XML Document Structure	2-40
Using the Expression Builder	2-44
Define Inventory Check Task	2-45

Defining Decisions	2-52
Decision: Has Order Been Confirmed?	2-52
Setting Up the XML Document Structure	2-56
Decision: Is Item Available?	2-59
Defining Events	2-62
Event: Cancellation Watch	2-63
Event: ShipBill Confirmation	2-69
Setting Up the Trigger	2-73
Setting Up the Workflow Template Definition Processing Properties	2-78
3. Defining the Start Order Processing Workflow	
Creating the Start Order Processing Workflow Template	3-2
Importing the Start Order Processing Workflow Template Definition	3-2
Defining the Start Order Processing Workflow	3-8
4. Defining the ShipBill Workflow	
5. Executing the Workflow Example	
Logging On to the Worklist Application	5-10
Executing the Workflow	5-11

About This Document

This document serves as a tutorial for the BEA WebLogic Process Integrator™ product. It steps you through the WebLogic Process Integrator Studio and Worklist applications, using a specific example to describe the process of defining and executing workflows.

This document is not a reference or a user guide and therefore does not cover all functions of the WebLogic Process Integrator product. For more information on any particular subject addressed in this document, refer to the *WebLogic Process Integrator Studio Guide* and the *WebLogic Process Integrator Worklist Guide*.

This document is organized as follows:

- Chapter 1, “Overview of the Example,” briefly describes the workflows in the example, illustrates how the workflows interact, and provides the main steps for defining and executing the workflow example.
- Chapter 2, “Defining the Order Processing Workflow,” provides procedures for creating and defining the Order Processing workflow, which processes the order through confirmation, inventor checking, and completion.
- Chapter 3, “Defining the Start Order Processing Workflow,” describes the Start Order Processing workflow, which is used to trigger the Order Processing workflow.
- Chapter 4, “Defining the ShipBill Workflow,” describes the Ship/Bill workflow, which handles the shipping and billing of the ordered item.
- Chapter 5, “Executing the Workflow Example,” provides procedures for manually executing the workflow example through the WebLogic Process Integrator worklist application.

What You Need to Know

This document is for users such as business analysts and workflow designers, who will use the WebLogic Process Integrator Studio to define and administer workflows. Chapter 5, “Executing the Workflow Example,” is for end users who will use the WebLogic Process Integrator Worklist to execute the workflow. This document assumes some familiarity with Java programming, XML documentation structure (XPath language structure), and business processes involved in the design of workflows.

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://e-docs.beasys.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 Process Integrator 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 Process Integrator 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 BEA WebLogic Process Integrator documents contain information that is relevant to using this product:

- *BEA WebLogic Process Integrator Tutorial*
- *BEA WebLogic Process Integrator Worklist Guide*
- *BEA WebLogic Process Integrator Installation and Configuration Guide*
- *BEA WebLogic Process Integrator Javadoc*
- *BEA WebLogic Process Integrator Release Notes*

Contact Us!

Your feedback on the BEA WebLogic Process Integrator 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 Process Integrator documentation.

In your e-mail message, please indicate which release of the BEA WebLogic Process Integrator documentation you are using.

If you have any questions about this version of BEA WebLogic Process Integrator worklist, or if you have problems installing and running the worklist, 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.
<i>italics</i>	Indicates emphasis or book titles.
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. <i>Examples:</i> <pre>#include <iostream.h> void main () the pointer psz chmod u+w * \tux\data\ap .doc tux.doc BITMAP float</pre>
<i>monospace</i> <i>italic</i> text	Identifies variables in code. <i>Example:</i> <i>String expr</i>
UPPERCASE TEXT	Indicates device names, environment variables, and logical operators. <i>Examples:</i> LPT1 SIGNON OR

Convention	Item
{ }	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. <i>Example:</i> 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.
...	Indicates one of the following in a command line: <ul style="list-style-type: none"> ■ 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. <i>Example:</i> 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.



1 Overview of the Example

The following sections provide an overview of the WebLogic Process Integrator workflow example:

- Description
- Important Terms

Description

The example presented in this document involves the following workflows:

- Order Processing Workflow
- Start Order Processing Workflow
- ShipBill Workflow

The example workflows interact by way of XML messaging and are supplied with the product on the WebLogic Process Integrator software CD and can be imported into your database. This document describes in detail how to define these workflows and in doing so describes the main functions of WebLogic Process Integrator.

About the Example

The example derives the flow from the confirmation of the order through shipping the item to and billing the customer. The front end handling the order is abstracted for the purpose of this exercise into an XML document, which provides the ordering information to the Order Processing workflow. (In a live situation, the XML document would be generated by a front-end customer application.)

The order process is split into two workflows: Order Processing, which processes the order, and ShipBill, which handles the shipping and billing of the order. The third workflow, Start Order Processing, is a workflow that has been created for this example. Its function is simply to start the order process. For the purposes of this example, the Start Order Processing workflow takes the place of the customer application.

It is important to note that this example could have been implemented as one workflow interaction with explicit calls to application functions through the use of several actions and without necessarily using XML.

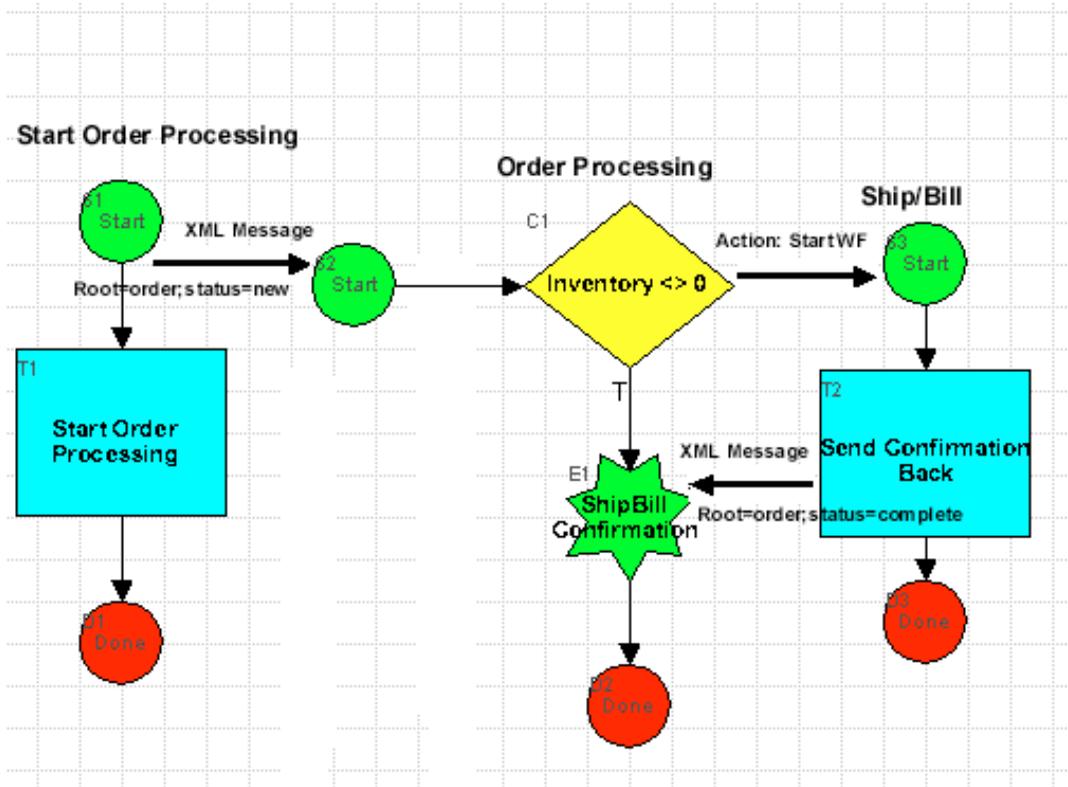
Example Workflows

The three workflows are as follows:

- **Start Order Processing**—This workflow contains a simple task, which creates the XML document, which in turn triggers the Order Processing workflow. In a “real” environment, this information could be captured by a front-end application and the information sent to the Process Integrator process engine through an XML document. In such a situation, a workflow would not simulate an application that sends an XML document representing the Order.
- **Order Processing**—This workflow takes the order and processes it through a first confirmation, inventory checking, and the completion of the order.
- **ShipBill**—This workflow handles the shipping and billing of the ordered item, and it is called by the Order Processing workflow. It therefore can be considered a sub-workflow to Order Processing.

The following diagram illustrates the interaction between the workflows defined in this tutorial.

Figure 1-1 Workflow Interaction



Important Terms

Before you get started with this tutorial, it is important to understand the following key terms:

- A workflow is a business process. Workflow automation is the automation of a business process, in whole or in part, during which information of any type is passed to the right participant at the right time according to a set of intelligent business rules that allow computers to perform most of the work while humans only have to deal with exceptions.
- A workflow template is a folder (or a container) in WebLogic Process Integrator Studio. This workflow template represents a workflow and is given a meaningful workflow name, such as Order Processing or Billing. The workflow template aggregates various definitions (or “versions”) of its implementation; these are referred to as workflow template definitions. Further, a workflow template is responsible for controlling which organizations can use the “contained” workflow template definitions.
- A workflow template definition is a definition (or “version”) of the workflow, distinguished by its Effective and Expiry dates. At run time, WebLogic Process Integrator starts an instance (or session) of a workflow template definition, selecting the most effective (or current and active) definition.

Main Steps

The main steps for defining and running the workflow example are as follows. These steps are described later in this document and in other documents.

1. Create the following user, role, and organization within WebLogic Server by adding them to the “WLPI Specific Section” of the `weblogic.properties` file:
 - User: joe
 - Role: Role1
 - Organization: ORG1

The default `weblogic.properties` file, provided on the WebLogic Process Integrator product CD, contains default WebLogic Process Integrator users, roles, and organizations. Simply copy an existing “WLPI” user, role, and organization and paste it in the appropriate section (`# WLPI users`, `# WLPI roles`, or `# WLPI orgs`), modifying the name to match those listed above. You can enter any password for the user.

Locate the `weblogic.properties` file in the following directory:

Drive: `/wlprocessintegrator/server/`

or for UNIX:

`\root\wlprocessintegrator\server\`

2. Activate the user, role, and organization that you added to the `weblogic.properties` file in WebLogic Process Integrator Studio following the instructions provided in the Chapter 2, “Administering Data within WebLogic Process Integrator,” in the *BEA WebLogic Process Integrator Studio Guide*.
3. Deploy the JavaBean POBean, which is used in the Order Processing workflow to check the inventory for the item ordered and return the results to the workflow. The file `POBean.jar` is provided on the product CD in the following directory:

Drive: `/wlprocessintegrator/studio/examples`

or for UNIX:

`\root\studio\examples`

The `weblogic.properties` file has the setup required to deploy this bean.

4. Import the workflow template definitions for Start Order Processing and ShipBill using the procedure described in “Importing the Start Order Processing Workflow Template Definition” in Chapter 3, “Defining the Start Order Processing Workflow.” The example discussed in this document involves three workflow template definitions, all of which are available in the following directory of the WebLogic Process Integrator CD:

```
Drive: /wlprocessintegrator/studio/examples
```

or for UNIX:

```
\root\studio\examples
```

You will follow the instructions provided in this guide to create and define the Order Processing workflow, so you do not need to import this workflow template definition. However, if you want to import all three example workflows and read this document with the workflows already set up and available in your database, you must import the workflows in the following order:

- a. Start Order Processing
- b. ShipBill
- c. Order Processing

Note: Importing the workflows in this sequence is only mandatory if you are importing all three workflows, rather than the suggested two workflows (Start Order Processing and ShipBill).

5. Define the workflows. (See Chapter 2, “Defining the Order Processing Workflow,” Chapter 3, “Defining the Start Order Processing Workflow,” and Chapter 4, “Defining the ShipBill Workflow.”)

You must redefine a few actions in the Order Processing and ShipBill workflows to resolve some external references. A warning message is displayed that the workflow contains some unresolved references. They are as follows:

- In Order Processing: In the decision:Inventory<>0, redefine the action Start Workflow in the True path of the decision.
- In the task Inventory Check, in the Activated tab of the Task Properties dialog box, redefine the action Perform Business Operation and choose the Check Inventory business operation.

1 *Overview of the Example*

- In ShipBill: In the task: Billing, in the Activated tab of the Task Properties dialog box, redefine the action Perform Business Operation and choose Calculate Total Price.
6. Execute the workflow. (See Chapter 5, “Executing the Workflow Example.”)

2 Defining the Order Processing Workflow

The following sections describe how to create and define the Order Processing workflow:

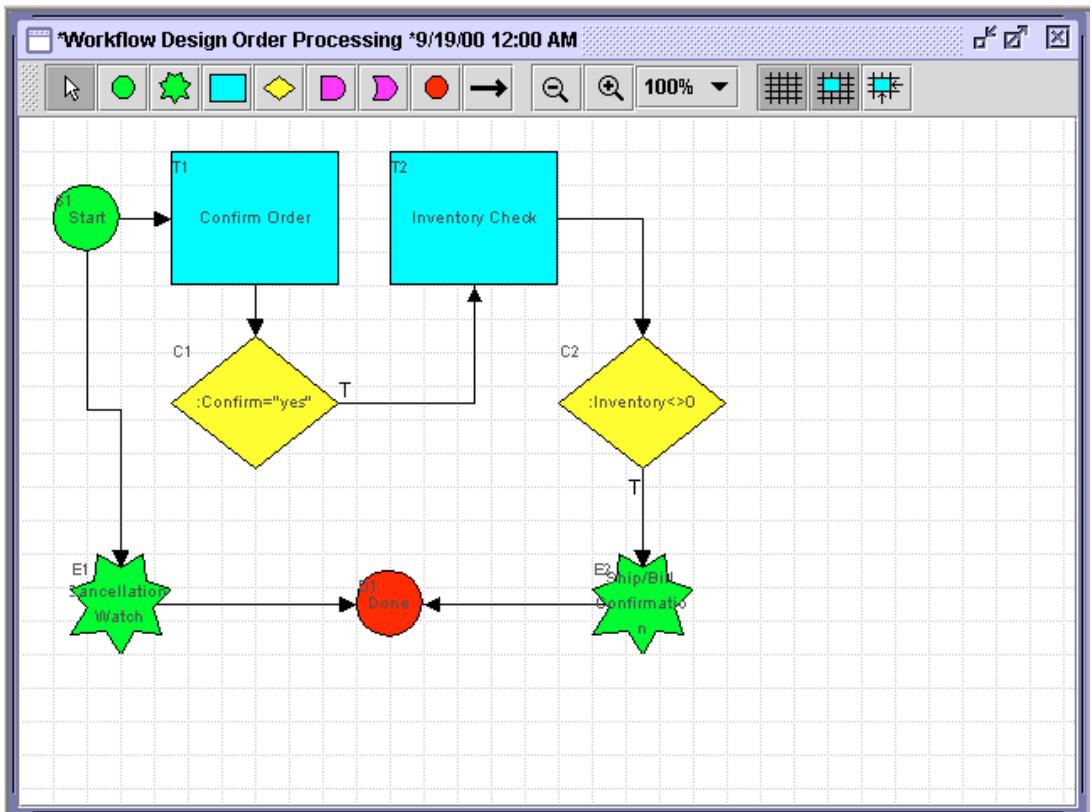
- Overview of the Order Processing Workflow
- Creating a New Workflow Template
- Creating a New Workflow Template Definition
- Drawing the Flow
- Defining Variables
- Defining Business Operations
- Defining Tasks
- Defining Decisions
- Defining Events
- Setting Up the Trigger
- Setting Up the Workflow Template Definition Processing Properties

Overview of the Order Processing Workflow

The following figure illustrates how the Order Processing workflow diagram should look after you define the entire workflow.

For details on the various workflow components represented by nodes in a workflow diagram, see Chapter 4, “Working with Workflow Components,” in the *BEA WebLogic Process Integrator Studio Guide*.

Figure 2-1 Order Processing Workflow



The Order Processing workflow will contain the following nodes:

Table 2-1 Order Processing Workflow Nodes

Icon	Node	Definition
	Start	XML event coming from workflow: Start Order Processing.
	Task	<p>Confirm Order</p> <ul style="list-style-type: none"> ■ Task is assigned to user: joe upon activation. ■ Upon execution, the user is presented with a prompt from the WebLogic Process Integrator Worklist application to validate the order. ■ The result: Yes or No is stored in the variable :Confirm. ■ Task is marked done.
	Decision	<p>Is order Confirmed? Does the variable Confirm= Yes?</p> <ul style="list-style-type: none"> ■ If Yes, flow moves on to the next task. ■ If No, a cancellation XML message is sent.
	Task	<p>Inventory Check</p> <ul style="list-style-type: none"> ■ Upon activation, the inventory bean is called. ■ The bean returns a value to the variable :Inventory. ■ The task is marked done.
	Decision	<p>Is stock available? Is variable Inventory greater than zero?</p> <ul style="list-style-type: none"> ■ If yes, the workflow: ShipBill is started and the flow moves on to the next node: Event: ShipBill Confirmation. ■ If No: The workflow is terminated.

2 Defining the Order Processing Workflow

Icon	Node	Definition
	Event	ShipBill Confirmation When the XML document that confirms the shipping and billing comes back from the ShipBill workflow, the ShipBill Confirmation event is triggered, and the Order Processing workflow is completed.
	Event	Cancellation Watch A cancellation event is triggered by an XML message directing the order to be cancelled. When this event is triggered, the Done node is reached and the workflow is completed.
	Done	Workflow Completed.

Creating a New Workflow Template

A workflow template is, in essence, a folder or a container for WebLogic Process Integrator workflow template definitions; each workflow template can hold one or more WebLogic Process Integrator workflow template definitions. Workflow template definitions are essentially different versions of a workflow.

WebLogic Process Integrator workflow template definitions are identified in the folder tree by an Effective and Expiry date; these are discussed in “Understanding Effective and Expiry Dates” of Chapter 3, “Defining and Maintaining Workflows” in the *BEA WebLogic Process Integrator Studio Guide*.

To create the Order Processing workflow template:

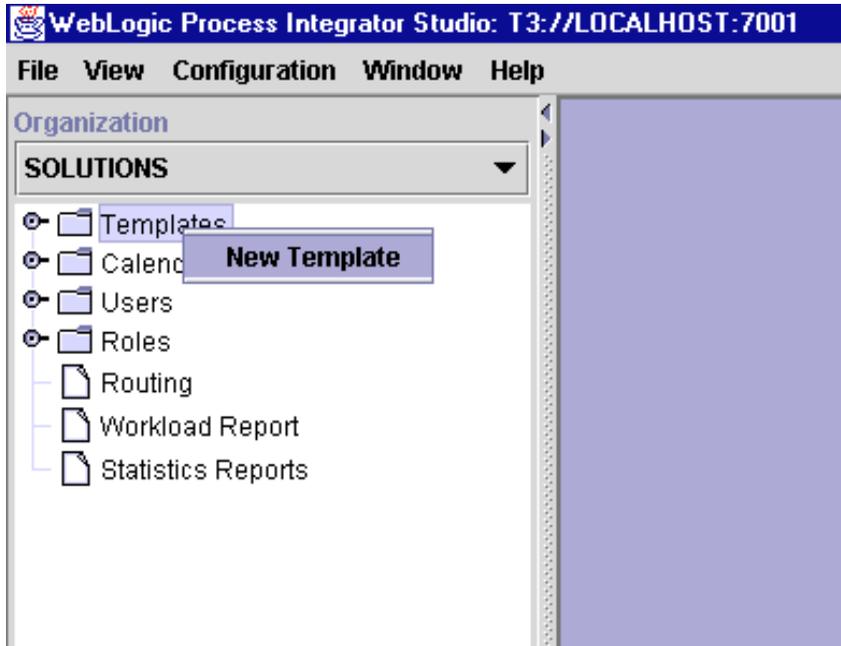
1. If you have not already done so, perform steps 1 and 2 in the “Important Terms” section of Chapter 1, “Overview of the Example.”
2. Start WebLogic Process Integrator Studio by selecting Start > Programs > WebLogic Process Integrator Studio to display the WebLogic Process Integrator Logon dialog box.

Figure 2-2 Logon to WebLogic Process Integrator Dialog Box



3. Enter your user ID, password, and the name of the WebLogic Process Integrator server to which you will connect, and click OK.
4. In the WebLogic Process Integrator Studio main window, create a new template by right-clicking Template and choosing New Template from the pop-up menu.

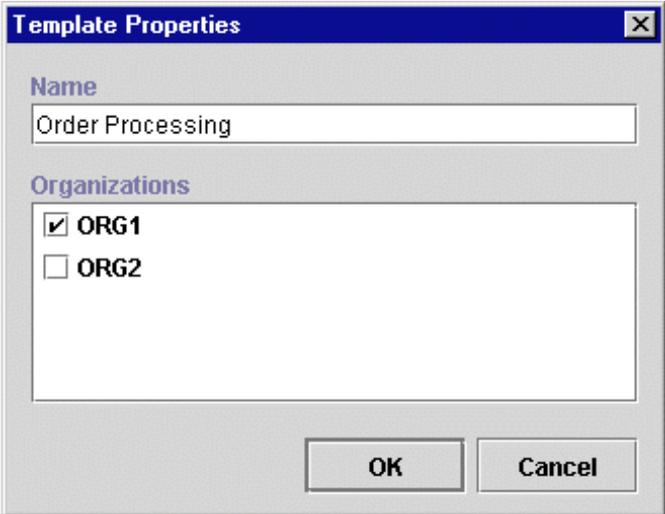
Figure 2-3 Choose New Template



5. In the dialog box that is displayed, enter Order Processing in the Name field and choose the organizations to which you want the template to belong: ORG1 in this example.

The Order Processing workflow template will be added under the Templates folder.

Figure 2-4 Template Properties Dialog Box

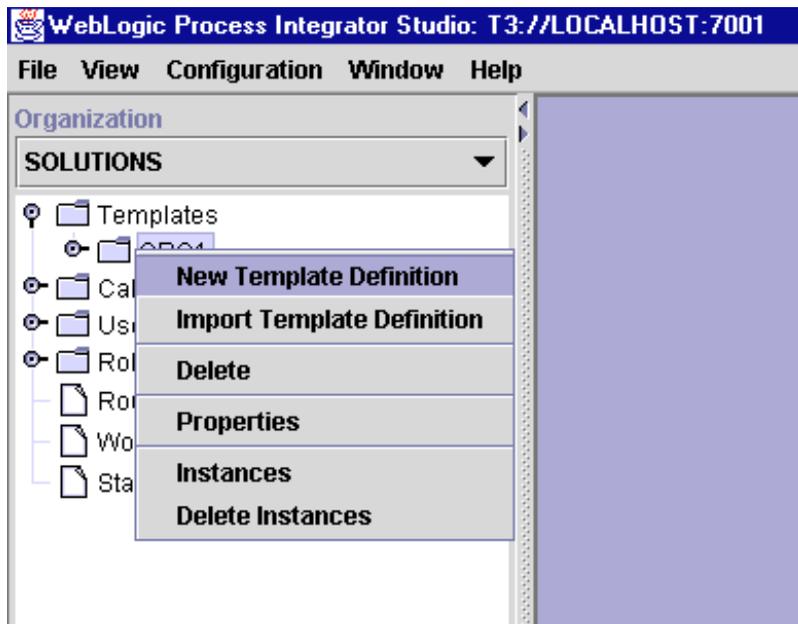


Creating a New Workflow Template Definition

To create a new workflow template definition for the new workflow template:

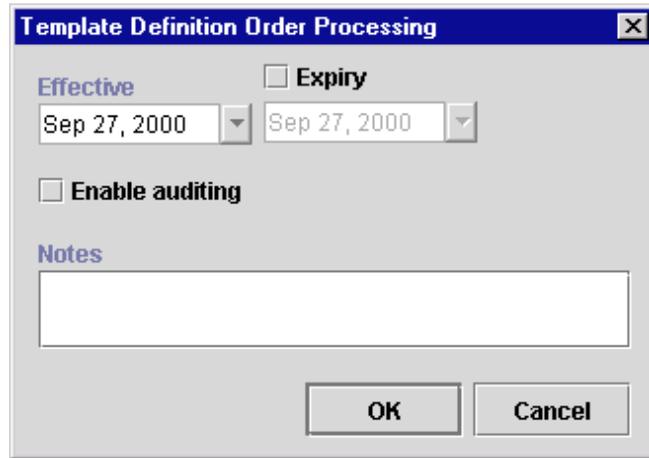
1. Under the Templates folder, right-click Order Processing workflow template and choose New Template Definition from the pop-up menu.

Figure 2-5 Choose New Template Definition



2. In the dialog box that is displayed, enter an Effective date for the new workflow template definition and click OK. (In this example, do not specify an Expiry date in order to make this workflow template definition always effective.)

Figure 2-6 Template Definition Dialog Box

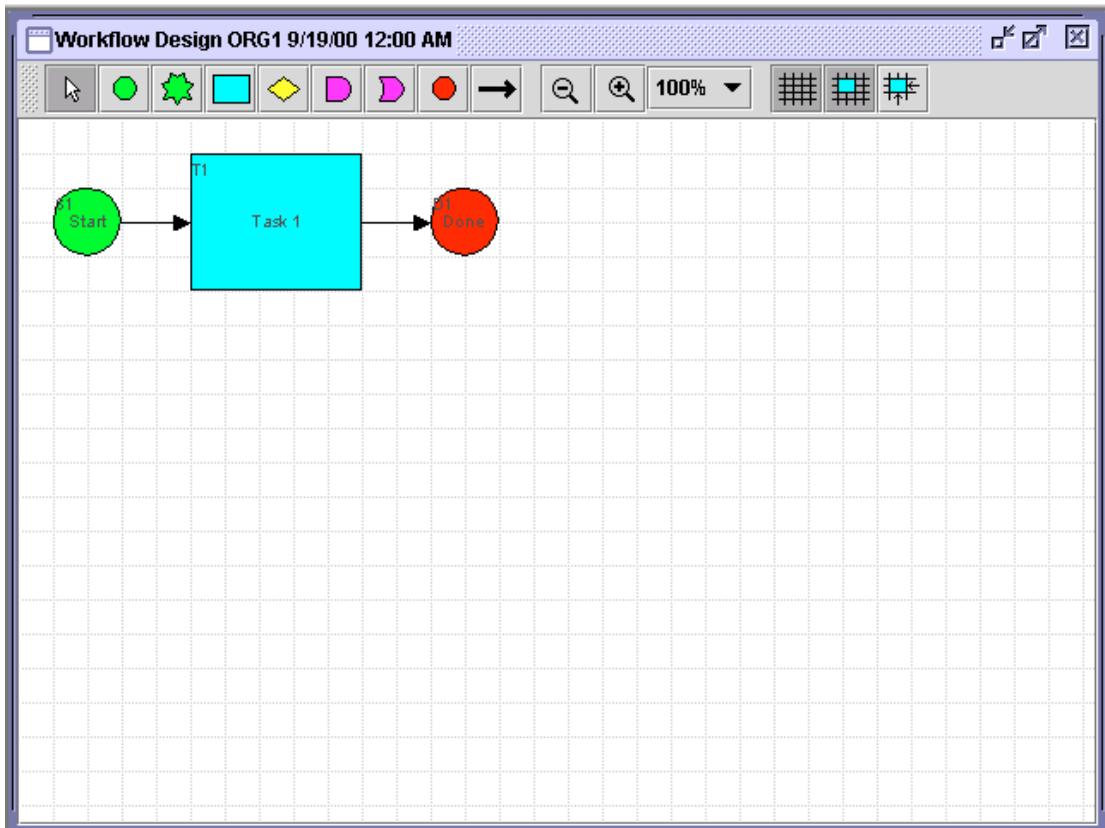


The Effective and Expiry dates allow you to make the workflow template definitions valid for a specific range of time. Each workflow template can contain one or more workflow template definitions, distinguished by the Effective and Expiry dates. At run time, the processing engine chooses the most effective workflow template definition for the specified workflow template, the most effective being the most current template definition that is marked active.

Drawing the Flow

Once you complete the Template Definition dialog box and click OK, the drawing area is displayed with a simple predefined workflow containing a start node, a task node, and a done node. From the toolbar, you choose the icons that represent nodes and other elements that you use to draw the flow. (Nodes in the drawing area are also referred to as shapes.)

Figure 2-7 Workflow Drawing Area



The Processing Order workflow requires the following nodes:

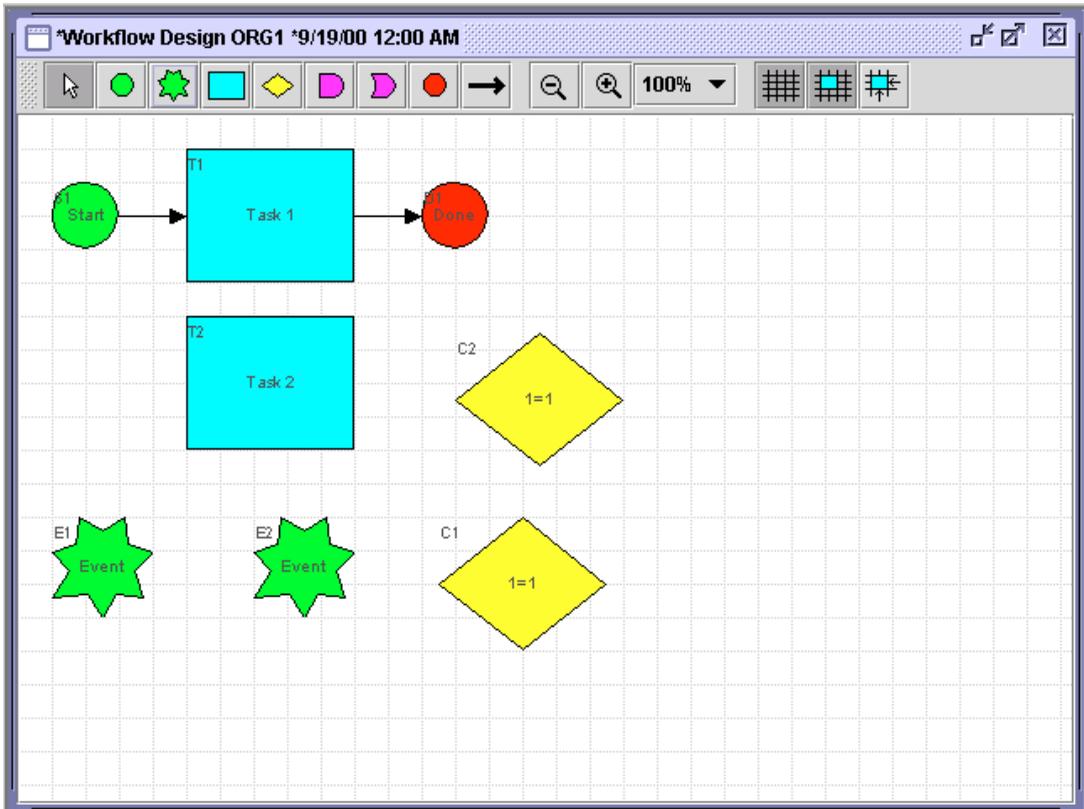
- Start
- Two tasks
- Two events
- Two decisions
- Done

Placing Node Icons (Shapes) in the Drawing Area

Place a shape in the drawing area by clicking its icon in the toolbar and then clicking the drawing area. This drops the shape (representing a workflow node) in the drawing area. Follow this procedure to place one task, two events, and two decisions in the drawing area. (The start and done nodes and one task node are already represented.)

You can move shapes within the drawing area by dragging and dropping them anywhere in the drawing area.

Figure 2-8 Shapes Placed in Drawing Area



Connecting the Nodes

Next, you connect the shapes in the drawing area to form the flow of the workflow. You connect shapes in one of two ways:

- Click the connector arrow in the icon toolbar to select it. Then, click the shape from which you want to connect. Continue to hold down your left mouse button, and drag the connector to the shape that you want to connect to the first shape. This inserts a line arrow between the two shapes; the shape of the arrow conforms to the positioning of the shapes in the drawing area.
- Double-click a shape in the drawing area to display its corresponding Properties dialog box. In the Next tab in the Properties dialog box, select the checkbox to the left of the shape name that should come next. Click OK.

To connect the nodes:

1. Before you begin making the connections, delete the connection between the Task 1 node and the Done node by clicking once on the arrow connecting them, right-clicking with your mouse, and choosing Delete Connection from the pop-up menu.
2. Make the following shape connections using the first connection method described at the beginning of this section.
 - a. Connect the Start node to Event 1 (E1). (The Start node, by default, is connected to Task 1 (T1). Leave this connection intact.)
 - b. Connect Event 1 (E1) to the done node.
 - c. Connect Task 1 (T1) to the C1 decision node.
 - d. Connect the C1 decision node to Task 2 (T2).

Note: A Decision node contains a condition, which can be either True or False. If the condition within the Decision node is verified as True, the workflow continues to Task 2.

- e. A dialog box appears asking you what kind of connection you want. Select True.
- f. Connect Task 2 (T2) to the C2 decision node.
- g. Connect the C2 decision node to Event 2 (E2).

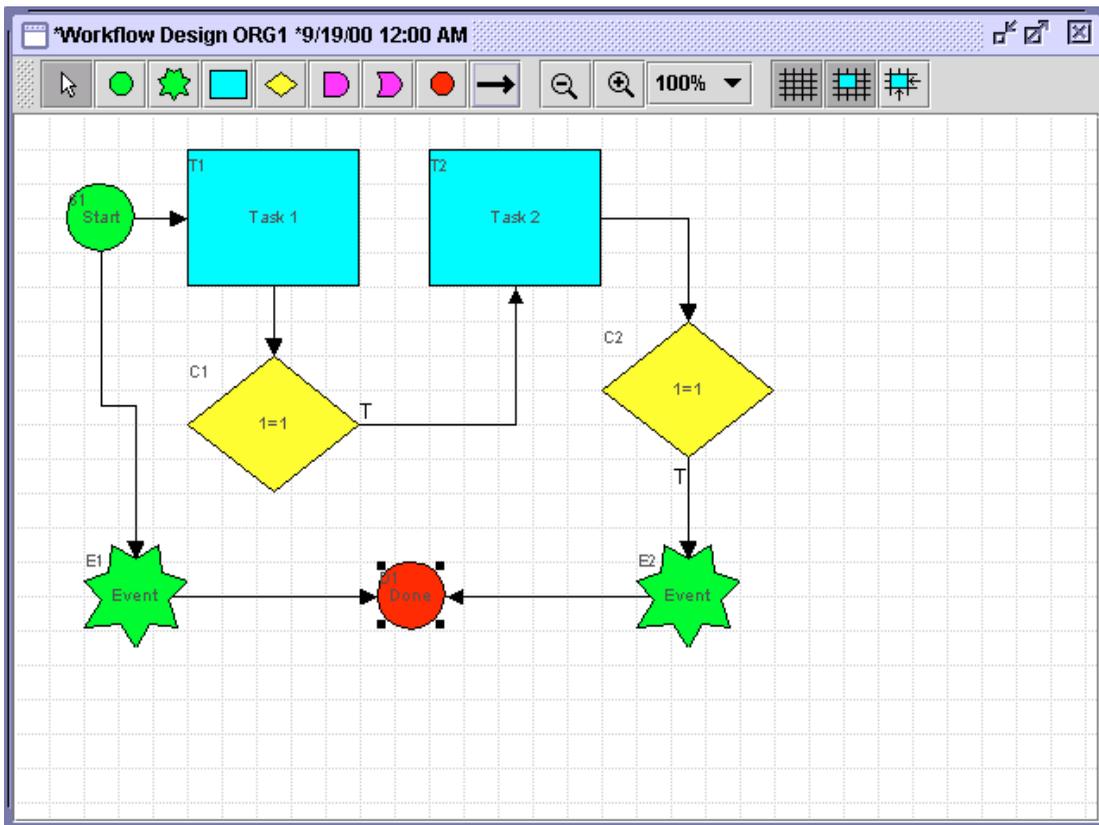
Note: A decision node contains a condition, which can be either True or False. If the condition within the decision node is verified as True, the workflow continues to Task 2.

h. A dialog box appears asking you what kind of connection you want. Select True.

i. Connect Event 2 (E2) to the done node.

You should have the following flow.

Figure 2-9 Completed Drawing



Renaming Nodes

Next, you will rename nodes in the drawing area.

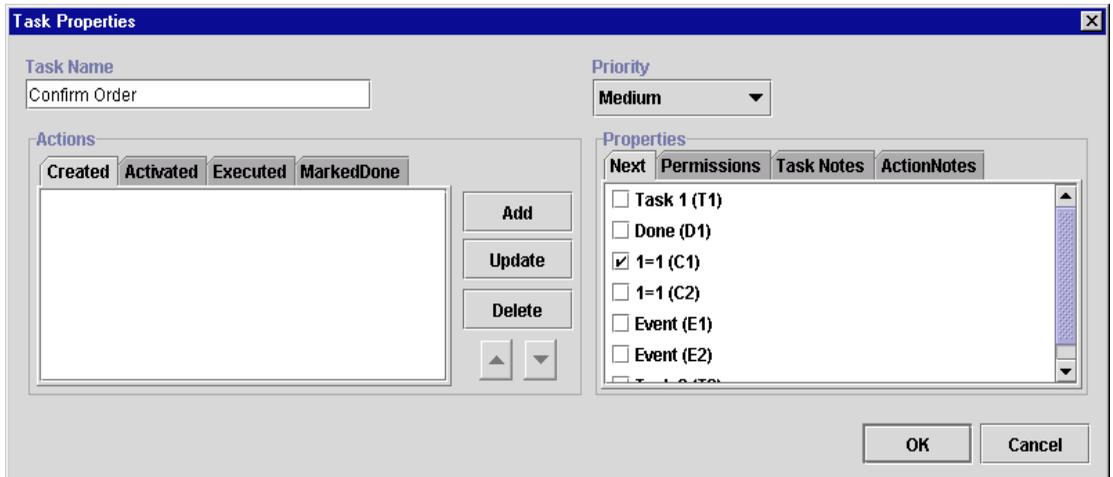
Task 1 (T1) Node

1. In the drawing area, double-click the Task 1 (T1) node to open the Task Properties dialog box.

This dialog contains the names of the nodes in the drawing area.

2. Enter Confirm Order in the Task Name field, and click OK.

Figure 2-10 Task Properties Dialog Box



Task 2 (T2) Node

1. In the drawing area, double-click the Task 2 (T2) node.
2. Enter Inventory Check in the Task Name field, and click OK.

Figure 2-11 Task Properties Dialog Box

The screenshot shows the 'Task Properties' dialog box. At the top, the title bar reads 'Task Properties'. Below the title bar, there are two main sections: 'Task Name' and 'Priority'. The 'Task Name' field contains the text 'Inventory Check'. The 'Priority' dropdown menu is set to 'Low'. Below these fields, there are two main panels: 'Actions' and 'Properties'. The 'Actions' panel has four tabs: 'Created', 'Activated', 'Executed', and 'MarkedDone'. Below the tabs is a large empty text area. To the right of this area are three buttons: 'Add', 'Update', and 'Delete', along with two small arrow buttons. The 'Properties' panel has four tabs: 'Next', 'Permissions', 'Task Notes', and 'ActionNotes'. Below the tabs is a list of tasks with checkboxes. The list includes: 'Task 1 (T1)', 'Done (D1)', '1=1 (C1)', '1=1 (C2)' (which is checked), 'Event (E1)', and 'Event (E2)'. At the bottom right of the dialog box are 'OK' and 'Cancel' buttons.

Event 1 (E1) Node

1. In the drawing area, double-click the Event 1 (E1) node to open the Event Properties dialog box.
2. Enter Cancellation Watch in the Task Name field, and click OK.

Figure 2-12 Event Properties Dialog Box

The screenshot shows the 'Event Properties' dialog box. The title bar is blue with the text 'Event Properties' and a close button (X). The main area is light gray. It contains the following sections:

- Description:** A text box containing 'Cancellation Watch'.
- Root Element:** An empty text box.
- Key Value Expression:** An empty text box with an 'A+BQ' button to its right.
- Condition:** An empty text box with an 'A+BQ' button to its right.
- Tabs:** Four tabs are visible: 'Variables' (selected), 'Next', 'Actions', and 'Notes'.
- Variables Table:** A table with two columns: 'Variable' and 'XML Expression'. The table is currently empty.
- Buttons:** To the right of the table are three buttons: 'Add', 'Update', and 'Delete'. At the bottom of the dialog are 'OK' and 'Cancel' buttons.

Event 2 (E2) Node

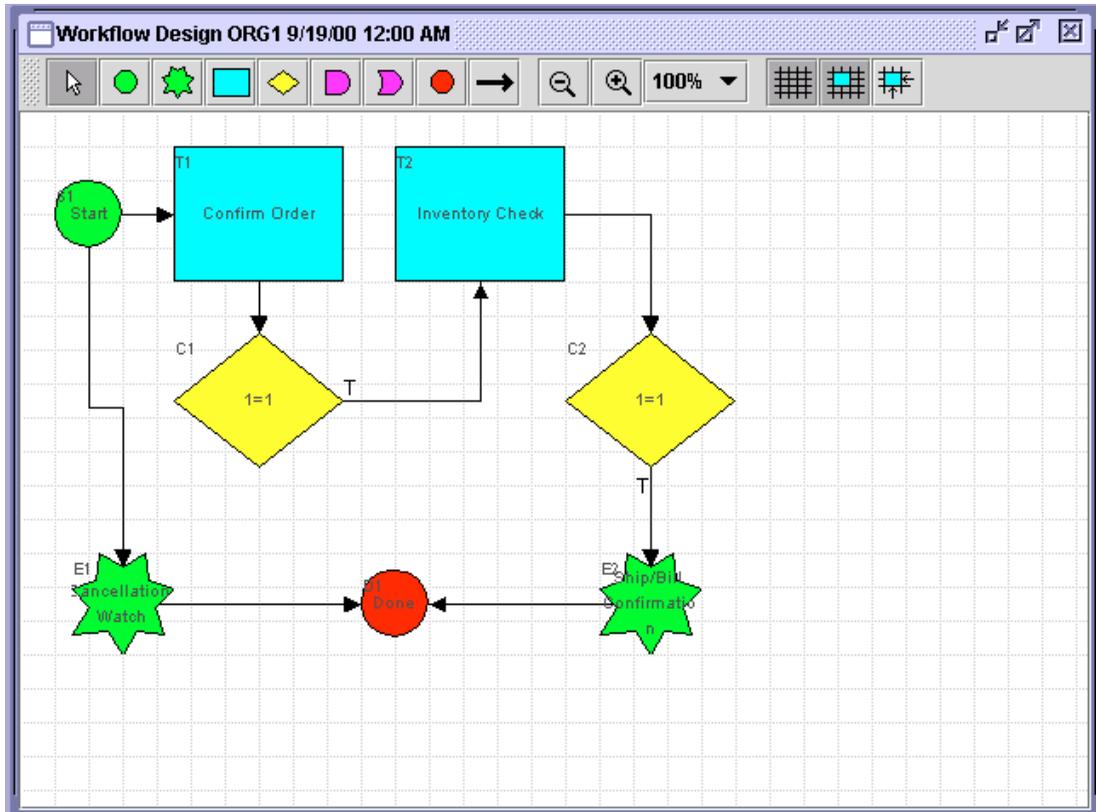
1. In the drawing area, double-click the Event 2 (E2) node.
2. Enter Ship/Bill Confirmation in the Task Name field, and click OK.

Figure 2-13 Event Properties Dialog Box

The dialog box is titled "Event Properties" and contains the following fields and controls:

- Description:** A text field containing "Ship/Bill Confirmation".
- Root Element:** An empty text field.
- Key Value Expression:** An empty text field with an "A+BQ" search icon to its right.
- Condition:** An empty text field with an "A+BQ" search icon to its right.
- Tabs:** "Variables", "Next", "Actions", and "Notes". The "Variables" tab is selected.
- Table:** A table with two columns: "Variable" and "XML Expression". The table is currently empty.
- Buttons:** "Add", "Update", and "Delete" buttons are located to the right of the table. "OK" and "Cancel" buttons are at the bottom right of the dialog.

Figure 2-14 Final Flow



After you connect all the nodes, you will define variables for the workflow; add the appropriate properties to each node; and define the actions for each node. These actions are executed at run time upon the instantiation of the workflow. Instantiation means that the workflow is started or placed into run time.

Defining Variables

Each workflow has a set of variables associated with it. A variable is typically used to store application-specific information required by the workflow at run time. This information is often used to control the logic within a workflow.

The Order Processing workflow requires the following variables.

Table 2-2 Variable Definitions

Variable	Type	Description
Confirm	String	User confirmation response; values are: Y or N
Inventory	Integer	Number of items available in inventory; can be zero
CustomerName	String	Name of the customer ordering the item
CustomerId	String	ID number of the customer ordering the item
CustomerMail	String	E-mail address of the customer
CustomerState	String	State or province for billing address and tax calculation
ItemName	String	Name of the ordered product
ItemNumber	Integer	Numerical identifier of the item
ItemQuantity	Integer	Quantity ordered
ItemPrice	Double	Unit price of the item
TotalPrice	Double	Total price of the order
OrderStatus	String	Status of the order
OrderNumber	Integer	Numerical identifier of the order

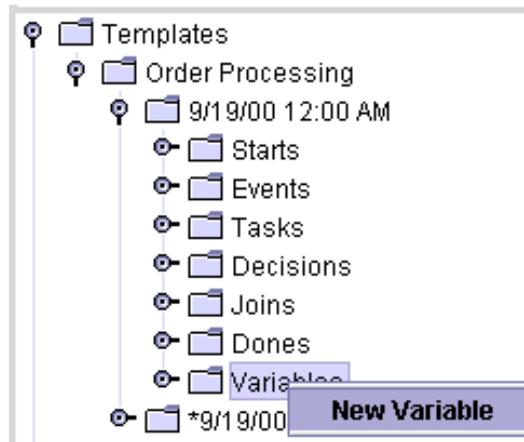
2 Defining the Order Processing Workflow

You will also need to define some of these variables for the ShipBill workflow that will be called by this workflow.

To define the variables for the Order Processing workflow:

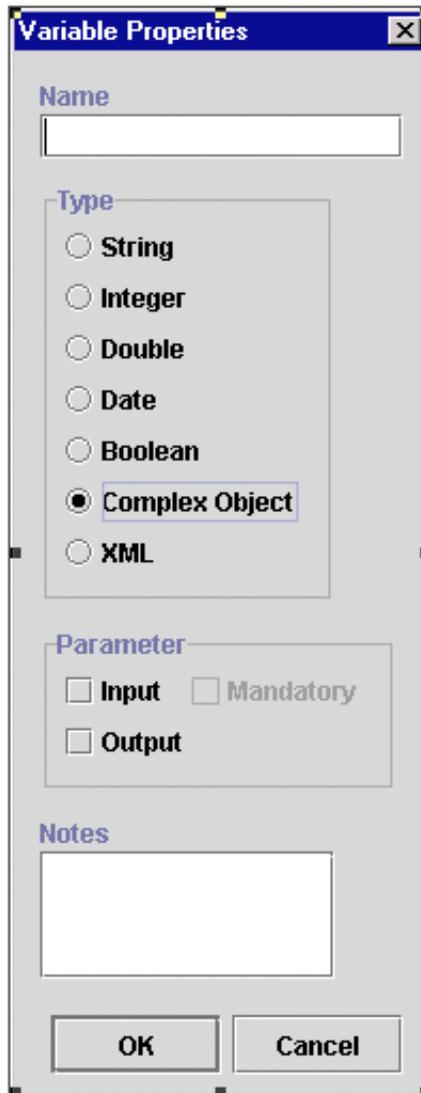
1. Expand the Order Processing folder, right-click Variable, and choose New Variable from the pop-up menu.

Figure 2-15 Choose New Variable



The Variable Properties dialog box is displayed.

Figure 2-16 Variable Properties Dialog Box



2 Defining the Order Processing Workflow

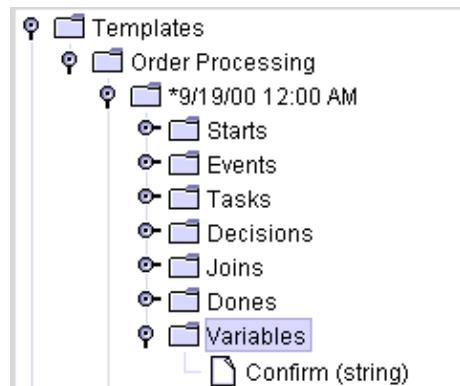
You will define all of the variables in the preceding table using the Variable Properties dialog box. For example, you will begin by defining the variable Confirm, which is a string type variable. It is a user confirmation response, and its possible values are Y (yes) or N (no).

Note: For now, you will only enter the variable name and type. You will define the possible values later in this tutorial.

To define this variable, proceed as follows:

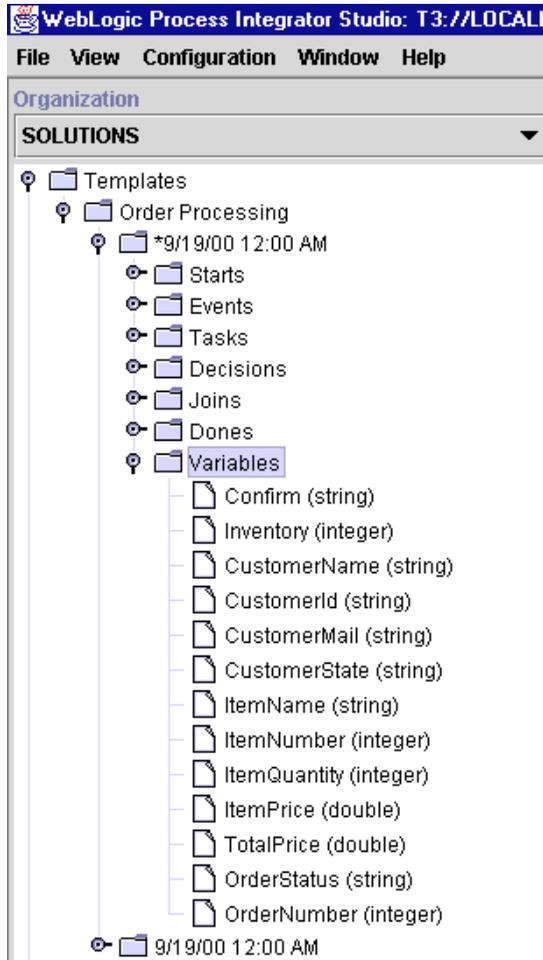
1. In the Variable Properties dialog box, enter the variable name Confirm in the Name field.
2. Select the String option button.
3. For this demonstration, do not enter values in the Parameter and Notes portions of the dialog box, and click OK.
4. Expand the Variables folder to view the new variable.

Figure 2-17 New Variable



5. Repeat these steps to create all variables listed in the preceding table. When you have created all new variables, your folder tree should look like the following:

Figure 2-18 New Variables List



Defining Business Operations

The WebLogic Process Integrator Business Operations facility enables you to display all Enterprise JavaBeans (EJBs) or Java classes that are registered in WebLogic Sever and their remote methods and parameters. Using this Business Operations facility, you provide a descriptive name for the remote methods and parameters of the EJBs or Java classes that you wish to deploy, thus providing non-technical users with an understanding of the business meaning behind these remote methods and parameters.

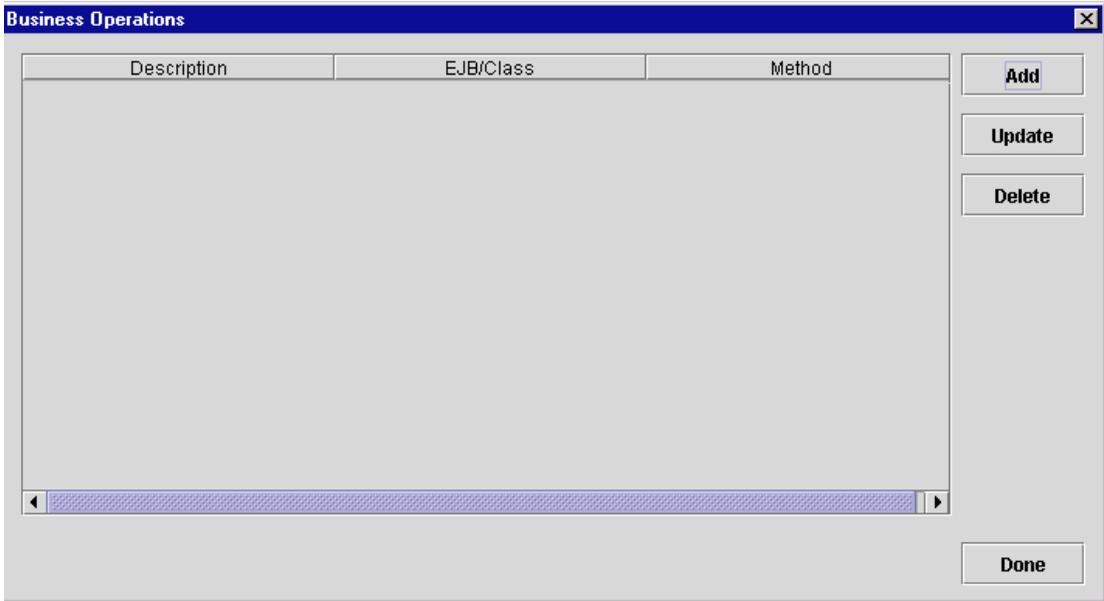
In WebLogic Process Integrator, a business operation represents a method call on an EJB or Java class instance. This method call is added to a workflow in WebLogic Process Integrator through use of the Perform Business Operation action. The invocation results of this action can be assigned to the workflow variables.

EJBs represent a component architecture for developing and deploying object-oriented, distributed, enterprise-level applications. The Business Operations feature, in effect, allows for the creation of customized actions, which create the connection between WebLogic Process Integrator and end-user applications and components.

To define business operations:

1. From the Configuration menu, choose Business Operation to display the Business Operations dialog box.

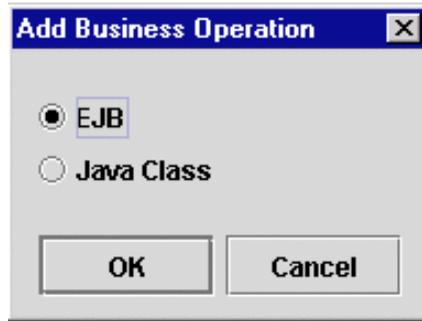
Figure 2-19 Business Operations Dialog Box



2 Defining the Order Processing Workflow

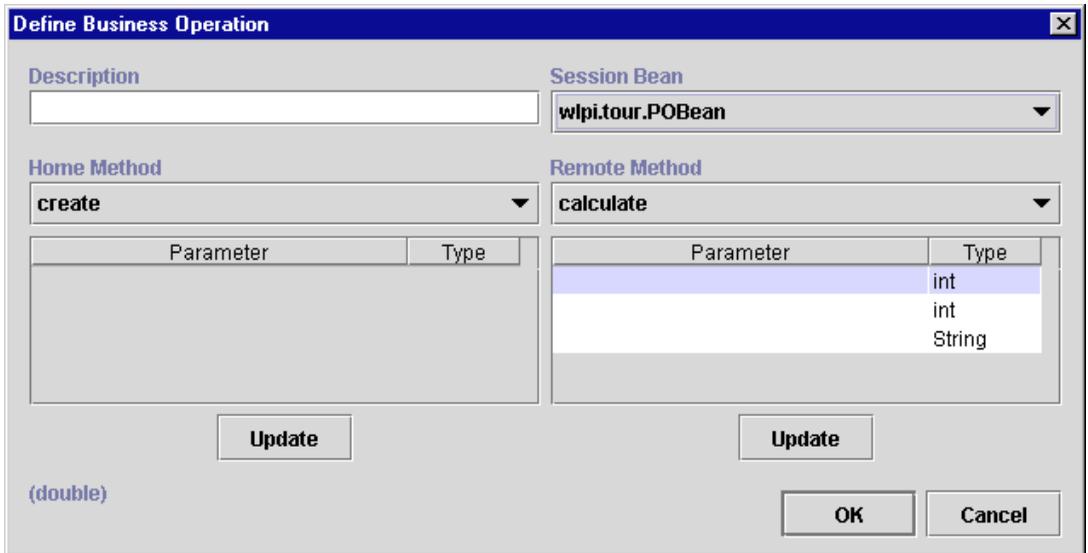
2. Click Add. You are prompted to add a new business operation.

Figure 2-20 Add Business Operation Dialog Box



3. Choose EJB and click OK. The Define Business Operation dialog box is displayed.

Figure 2-21 Define Business Operation Dialog Box



The Session Bean drop-down list displays all registered JavaBeans that are defined for BEA WebLogic Server. For each JavaBean, the methods are displayed in the Remote Method drop-down list, and for each method, the parameter types are displayed under this list. You will provide a meaningful name for each parameter.

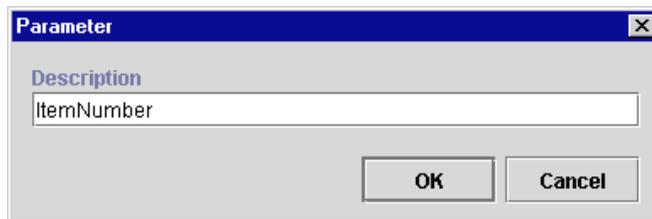
Using the Define Business Operation dialog box, you will define two business operations. These business operations represent remote methods of invoking the `wlpi.tourPOBean` JavaBean. This JavaBean is used in the workflow to check the inventory for the item ordered and return the results to the workflow.

The business operations are:

- The method `CheckInventory`, called from the Order Processing workflow, returns the number of items available in stock.
- The method `Calculate` calculates the final price of the order and is called within the workflow `ShipBill`.

4. From the Session Bean list, select `wlpi.tour.POBean`.
5. In the Description field, enter `Check Inventory`.
6. Choose `CheckInventory` from the Remote Method drop-down list.
7. Highlight the Parameter field, and click `Update` to display the Parameter dialog box.

Figure 2-22 Parameter Dialog Box



8. In the Description field, enter the parameter description `CheckInventory`, and click `OK`.

Note: The parameter type is displayed for each parameter, because these parameters are already defined.

Figure 2-23 Define Business Operation Dialog Box

Define Business Operation

Description
Check Inventory

Session Bean
wipi.tour.POBean

Home Method
create

Remote Method
checkInventory

Parameter	Type
ItemNumber	int

(int)

Update Update OK Cancel

Next you will define the business operation Calculate.

9. In the Description field of the Define Business Operation dialog box, enter Calculate Total Price.
10. Choose Calculate from the Remote Method drop-down list.
11. In the Parameters column, highlight the first parameter, which is of the type integer, and click Update to display the Parameter dialog box.
12. In the Description field, enter the parameter description ItemNumber, and click OK.
13. Repeat these steps to enter the remaining two parameter names. The second parameter should be named ItemQuantity, and the third parameter should be named CustomerState.

Figure 2-24 Define Business Operation Dialog Box

Define Business Operation

Description
Calculate Total Price

Session Bean
wlpi.tour.POBean

Home Method
create

Remote Method
calculate

Parameter	Type
ItemNumber	int
ItemQuantity	int
CustomerState	String

(double)

Update Update OK Cancel

The Business Operations dialog box now displays the two defined business operations.

Figure 2-25 Business Operations Dialog Box

Business Operations

Description	EJB/Class	Method
Check Inventory	wlpi.tour.POBean	checkInventory
Calculate Total Price	wlpi.tour.POBean	calculate

Once the actual business operations are defined, you can call them into the workflow using the Perform Business Operation action. This action calls the business operations into the workflow.

Instructions for using the Perform Business Operations action are provided in “Define Inventory Check Task” and in Chapter 4, “Defining the ShipBill Workflow.”

Defining Tasks

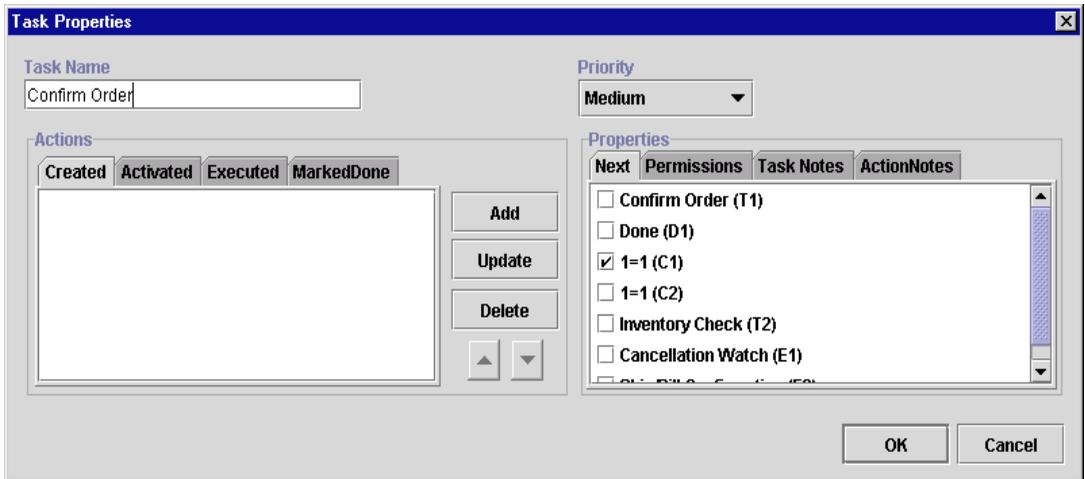
A workflow consists of a series of tasks, each of which represents a discrete activity to be performed automatically by the WebLogic Process Integrator server or by a workflow end user by using the worklist application. In this section, you define the tasks in the Order Processing workflow.

Defining the Confirm Order Task

To define the Confirm Order task:

1. In the workflow diagram, double-click the task Confirm Order to display the Task Properties dialog box.

Figure 2-26 Task Properties

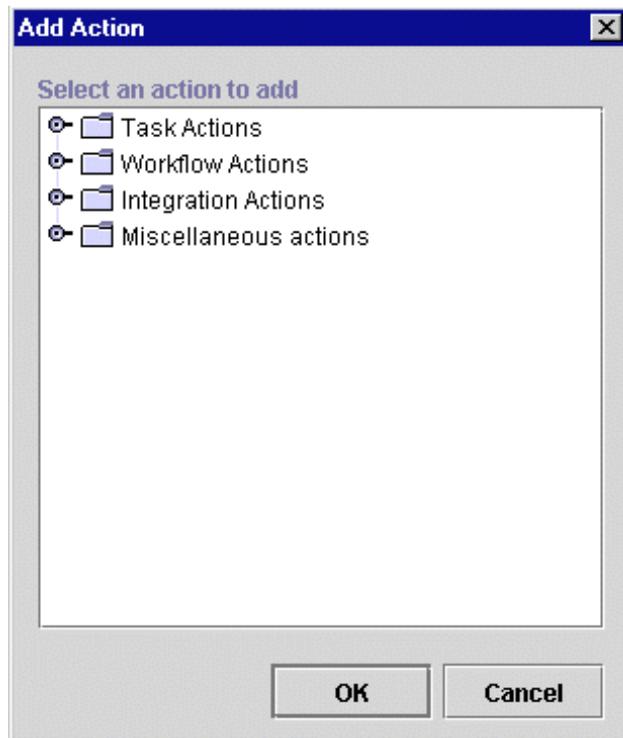


Confirm Order is the first task in this workflow. At run time, this task is performed by an end user. Therefore, it is assigned to an actual user and is displayed on that user's worklist upon instantiation of the workflow.

The tabs on the left side of the dialog box display four separate lists of actions for the tasks that are performed at the four task events. A task responds to an event at run time by executing the associated actions. The four task events are:

- Created—The Created event is automatically triggered when the workflow is started.
 - Activated—The flow hits this particular task and the task is available to be performed.
 - Executed—An event occurs that causes the task to be executed. This usually occurs when the user double-clicks on the task in a worklist.
 - Marked Done—The Marked Done event is automatically raised when the task is marked done. This can be done through the Mark Task as Done action or through an API call.
2. Select the Activated tab. The Assign Task "Confirm Order" to User WorkflowAttribute("Initiator") action, by default, is already defined in this task event.
 3. Select the Assign Task to User action in the Activated tab, and click Delete to delete this action.
 4. Click Yes to confirm the delete.
You are now ready to add the new Assign Task to User action.
 5. Click Add to display the Add Action dialog box.

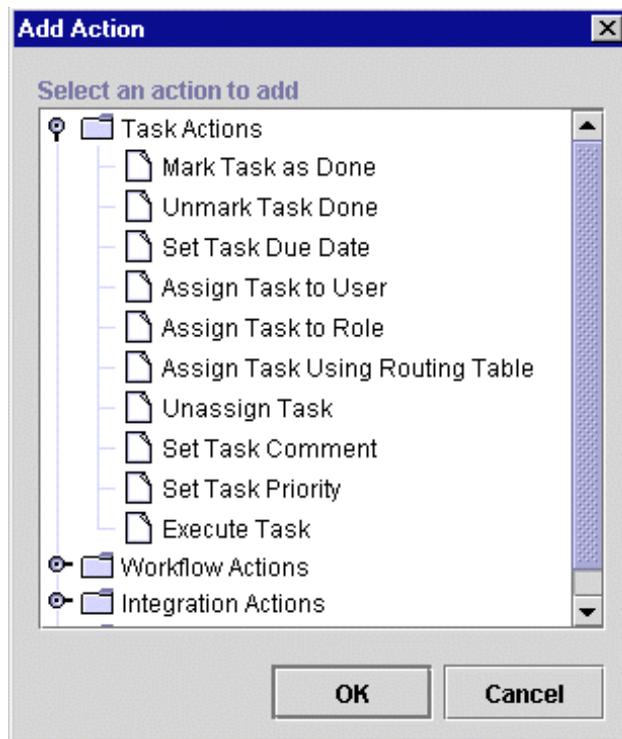
Figure 2-27 Add Action Dialog Box



Actions are the basic building blocks of the workflow. They can be defined within tasks, events, decisions, starts, and as a sub-action of an action itself.

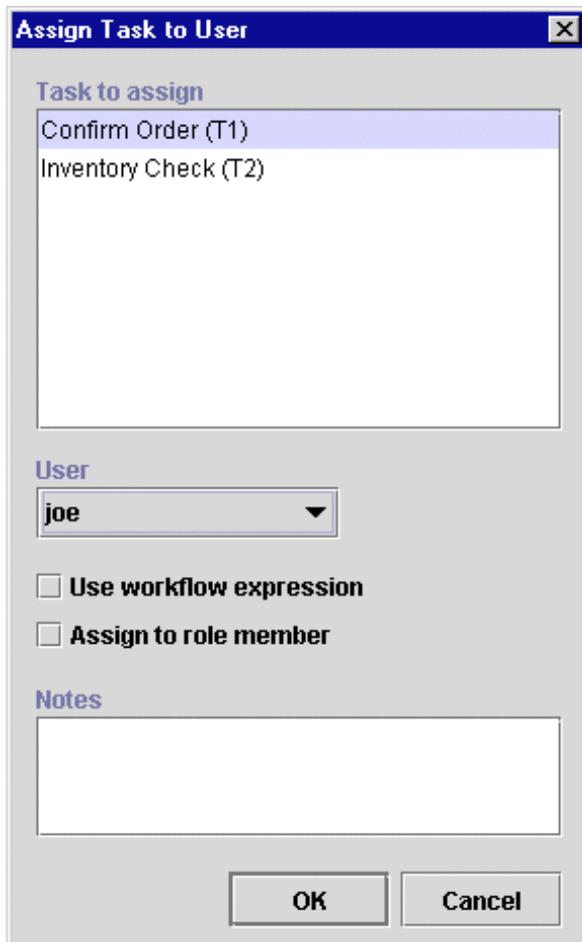
The action categories are Task, Workflow, Integration, and Miscellaneous. The task category contains all actions relevant to tasks, including actions that allow the assignment of tasks to users and roles.

Figure 2-28 Add Action Dialog Box



6. Select Assign Task to User and click OK.

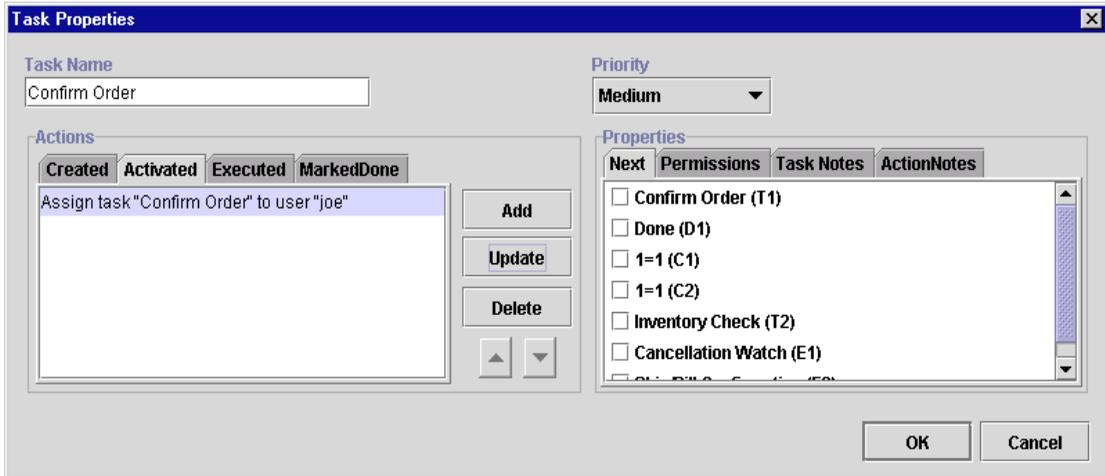
Figure 2-29 Assign Task to User Dialog Box



7. In the Task to Assign portion of the dialog box, select the task Confirm Order.
8. Select joe from the User drop-down list, and click OK.

The Assign Task to User action is displayed in the Activated tab.

Figure 2-30 Task Properties: Assign Task to User Action

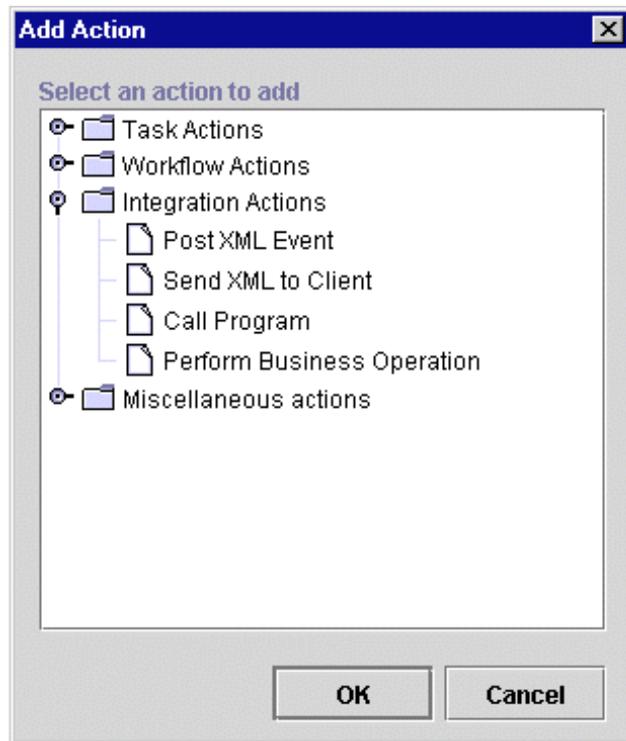


The Confirm Order task, upon execution, prompts the user to confirm the order. The Decision node that succeeds the task directs the flow according to the user's response.

9. Select the Executed tab of the task and click Add.
10. Select Integration Actions from the Add Action dialog box.

The action that allows the user to interact with the worklist and enter a response to a question or value for a variable is Send XML to Client in the Integration Actions category.

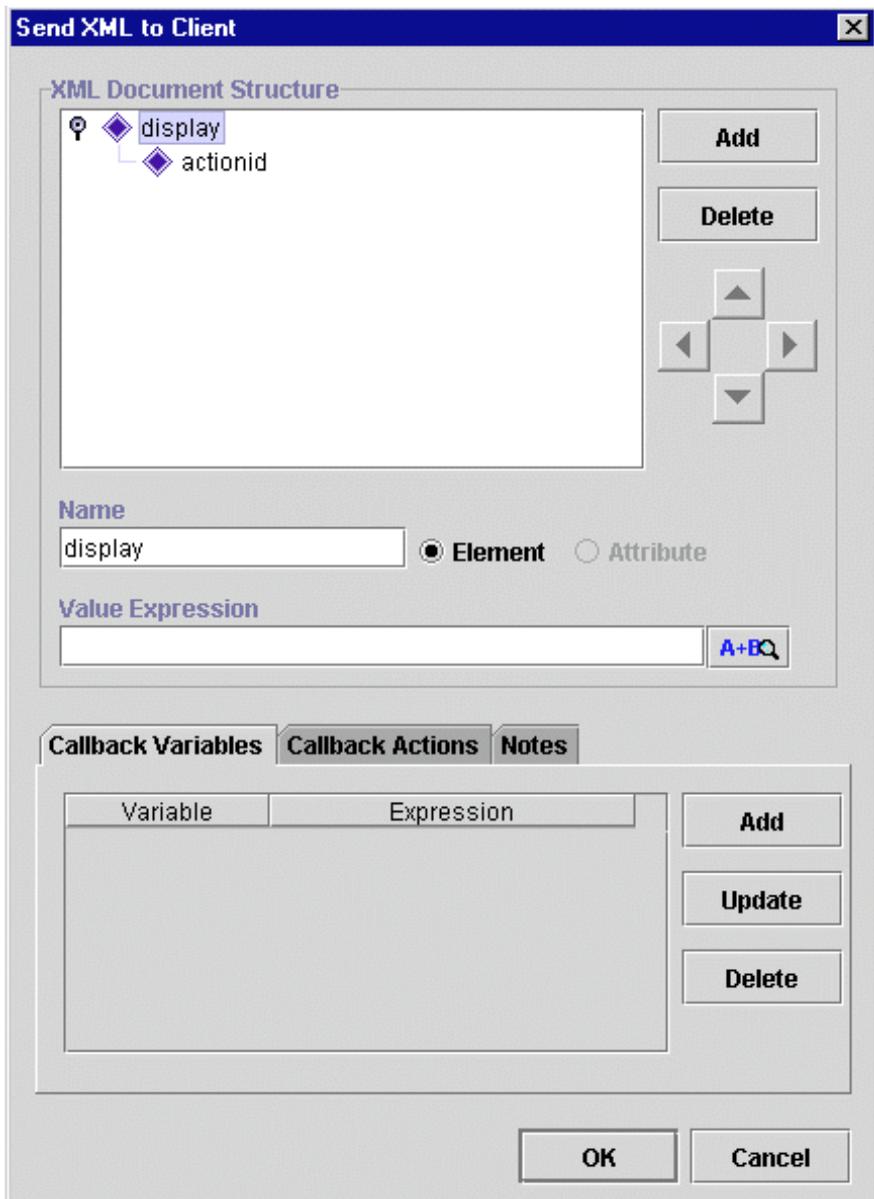
Figure 2-31 Integration Actions



11. Select Send XML to Client, and click OK to display the Send XML to Client dialog box.

Proceed to the next section, “Defining the Order Processing Workflow,” to continue defining the Confirm Order task.

Figure 2-32 Send XML to Client Dialog Box



Understanding the XML Document Structure

In the Send XML to Client dialog box, you define an XML document, which will be sent out to the client application. The client application should be programmed to identify this XML document, respond to it appropriately, and send an XML document to the WebLogic Process Integrator server with information that can be used to populate workflow variables. The WebLogic Process Integrator Worklist responds to certain default setups of this action.

Default structures for the XML document can be used to:

- Prompt the user for variable input.
- Display a message-box to the user containing information or asking a question. (You can set up the responses to be OK, OK/Cancel, Yes/No, or Yes/No/Cancel.)
- Call a program on the client machine.
- Call an API.

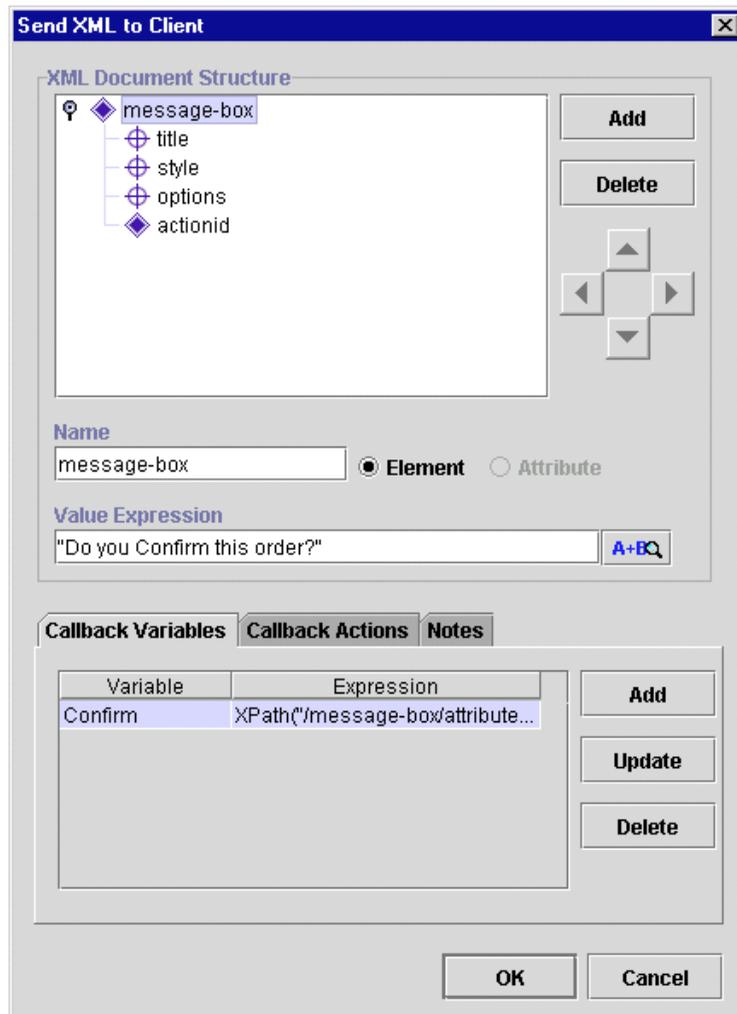
To prompt the user for a simple Yes/No confirmation, the XML document must have the following structure, where:

- The root element of the XML document is `message-box`. The value of `message-box` is the actual question that will be displayed to the user.
- The root element `message-box` has an attribute `title` that is the title of the message-box to be displayed to the user.
- The root element `message-box` has an attribute `style` that is the type of message-box that will be displayed: `plain`, `information`, `question`, `warning`, or `error`. In this case, the `style` will be a question.
- The root element `message-box` has an attribute `options` that is the admissible response type: `OK`, `OK/Cancel`, `Yes/No`, or `Yes/No/Cancel`. In this case, the `options` will be `Yes/No`.
- `actionid` is an internal unique ID that identifies this particular Send XML to Client action and will also identify and match the message coming back from the worklist.

Defining the XML Document Structure

Refer to the following figure and procedure to define the XML document structure. The figure shows how the Send XML to Client dialog box should look after you complete the procedure.

Figure 2-33 Final XML Document Structure



To define the XML document structure:

1. In the Send XML to Client dialog box, change the default root element default to message-box by highlighting the default root element and entering the name message-box in the Name field.
2. In the Value Expression field, enter the following value: "Do you Confirm this Order?" This is the question that will be presented to the end user in the message-box.
3. Select the root element message-box and click Add to add the attribute title to message-box. A new XML node is displayed in the XML Document Structure portion of the dialog box.
4. Select the Attribute radio button to make this new XML node an attribute, and rename the new XML node by entering title in the Name field.
5. In the Value Expression field, enter the message-box title: "Confirm Order".
6. Select the root element message-box and click Add to add the attribute style to message-box. A new XML node is displayed in the XML Document Structure portion of the dialog box.
7. Select the Attribute radio button to make this new XML node an attribute, and rename the new XML node by entering style in the Name field.
8. In the Value Expression field, enter the message-box style: "question".
9. Select the root element message-box and click Add to add the attribute options to message-box. A new XML node is displayed in the XML Document Structure portion of the dialog box.
10. Select the Attribute radio button to make this new XML node an attribute, and rename the new XML node by entering options in the Name field.
11. In the Value Expression field, enter the message-box options: "Yes_No".
12. Using the up and down arrows, rearrange the XML nodes so that they appear in the XML document structure in the following order:
 - title
 - style
 - options
 - actionid

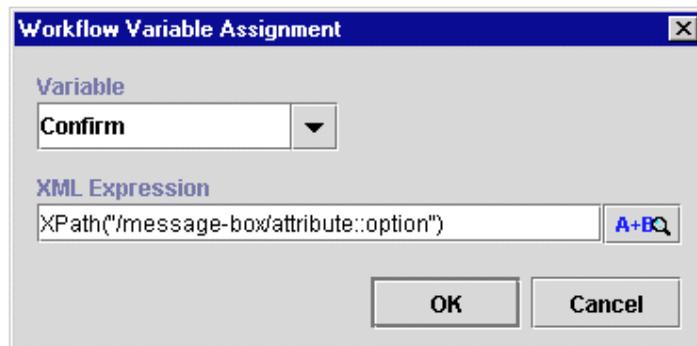
13. Click the Add button at the bottom of the dialog box to add a callback variable. The Workflow Variable Assignment dialog box is displayed.
14. From the Variable drop-down list, select the Confirm variable.
15. In the XML Expression field, enter the XML expression for the message-box using the XPath language structure. The XML expression for Confirm should be:

```
XPath("/message-box/attribute::option")
```

Note: If you know XML, you can type the expression directly into this field. Otherwise, click the A+B button to access the Expression Builder. For instructions on using the Expression Builder, see Chapter 6, “Using Expressions and Conditions,” of the *BEA WebLogic Process Integrator Studio Guide* and “Using the Expression Builder” in this document.

BEA WebLogic Process Integrator uses the XPath XML model. For information on XPath, refer to the following Web site: <http://www.w3.org/TR/xpath.html>. Also, refer to Chapter 6, “Using Expressions and Conditions,” in the *BEA WebLogic Process Integrator Studio Guide*.

Figure 2-34 Workflow Variable Assignment



In the example here, the user’s response will be stored in the workflow variable Confirm as defined in the Callback Variables tab in the lower part of the Send XML to Client dialog box.

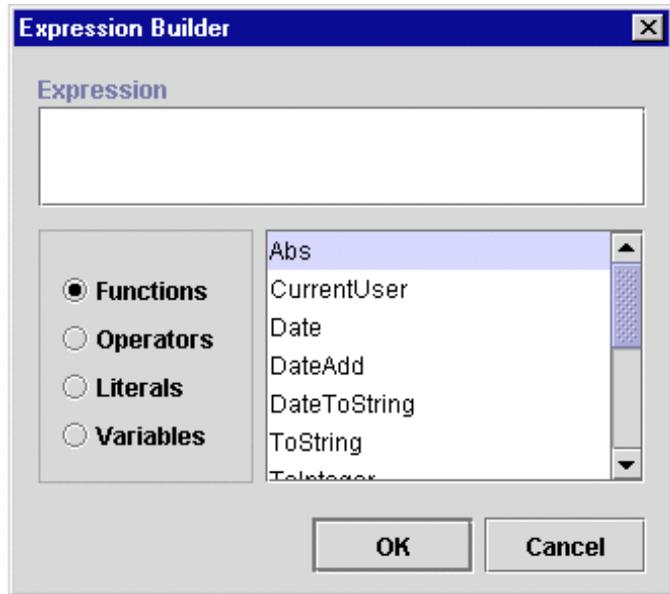
16. Click OK.

Using the Expression Builder

You use the Expression Builder to define the value of the variable coming back in an XML document from the worklist client application. The function XPath is used to parse the XML document and extract the relevant value from it.

The Expression Builder, available in dialog boxes where you see A+B, enables you to build expressions, by using predefined functions, literals and operators, which will be evaluated and executed at run time. Workflow variables are usually used in expressions. Clicking A+B in any dialog box displays the Expression Builder dialog box.

Figure 2-35 Expression Builder



See Chapter 6, “Using Expressions and Conditions,” of the *BEA WebLogic Process Integrator Studio Guide* for detailed explanations of each function available in the Expression Builder.

Define Inventory Check Task

Upon activation, the Inventory Check task calls the JavaBean POBean, which checks the inventory for the item ordered and returns the results to this workflow.

To define the Inventory Check task:

1. In the workflow diagram, double-click the Inventory Check task.
2. In the Task Properties dialog box, select the Activated tab, then click Add.
3. In the Add Action dialog box under Integration Actions, select the Perform Business Operation action, and click OK to display the Perform Business Operation dialog box.

Figure 2-36 Add Action: Perform Business Operation

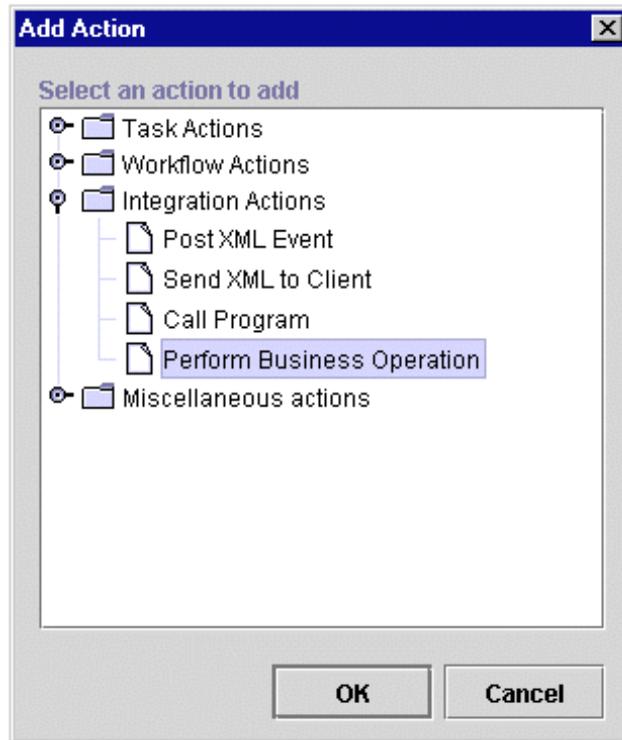
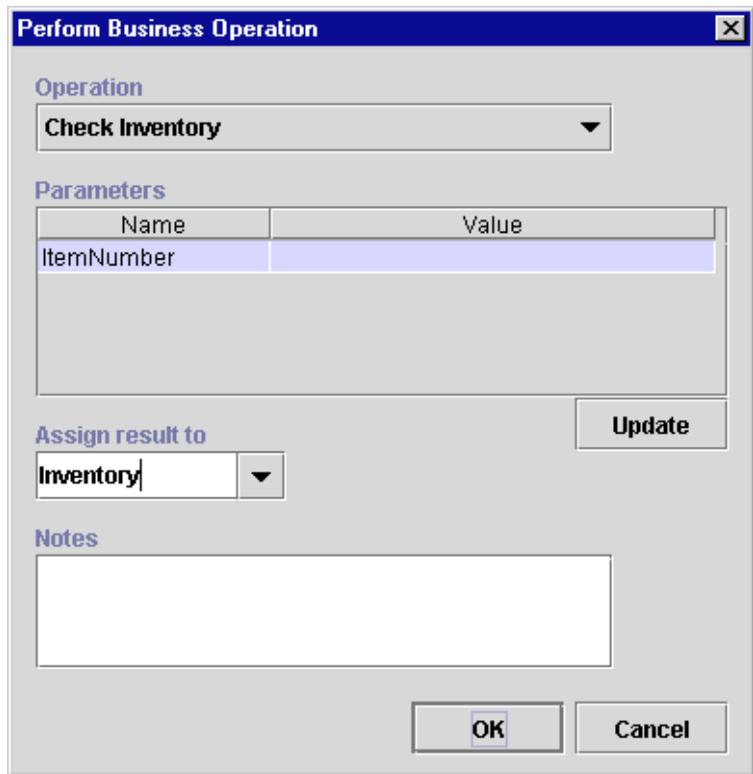


Figure 2-37 Perform Business Operation Dialog Box



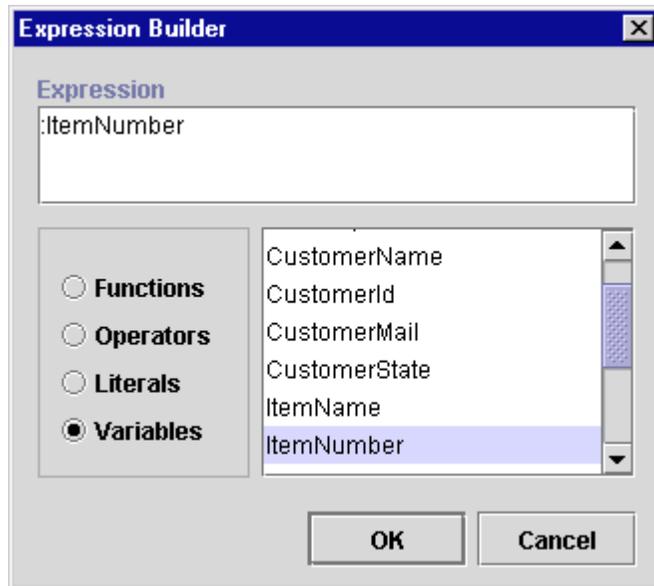
The dialog box titled "Perform Business Operation" has a blue title bar with a close button (X). It contains the following sections:

- Operation:** A drop-down menu with "Check Inventory" selected.
- Parameters:** A table with two columns: "Name" and "Value". The row "ItemNumber" is highlighted.
- Assign result to:** A drop-down menu with "Inventory" selected.
- Notes:** A large empty text area.
- Buttons:** "Update" (top right), "OK" (bottom center), and "Cancel" (bottom right).

Name	Value
ItemNumber	

4. The Operation drop-down list contains the business operations CheckInventory and Calculate that you defined earlier. Select CheckInventory.
5. In the Parameters section, the variable ItemNumber (defined earlier) is automatically inserted. Highlight it and click the Update button to display the Expression Builder. You will use the Expression Builder to specify the value of the ItemNumber input parameter.

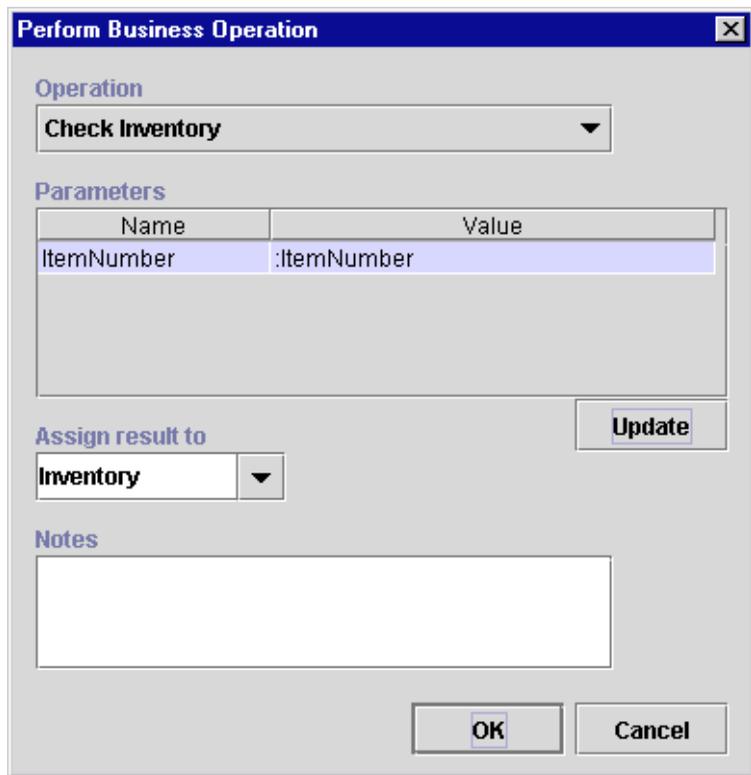
Figure 2-38 Expression Builder Dialog Box



6. Select the Variables radio button, and from the scrollable list of variables (which you defined earlier), select the ItemNumber input parameter.
7. Double-click on this parameter to display it in the Expression field, then click OK.

The Value field of the Perform Business Operation dialog box is now populated with the `:ItemNumber` parameter value.

Figure 2-39 Perform Business Operation Dialog Box



The dialog box is titled "Perform Business Operation" and contains the following sections:

- Operation:** A dropdown menu with "Check Inventory" selected.
- Parameters:** A table with two columns: "Name" and "Value". The first row contains "ItemNumber" and ":ItemNumber".
- Assign result to:** A dropdown menu with "Inventory" selected.
- Notes:** An empty text area.
- Buttons:** "Update", "OK", and "Cancel".

Name	Value
ItemNumber	:ItemNumber

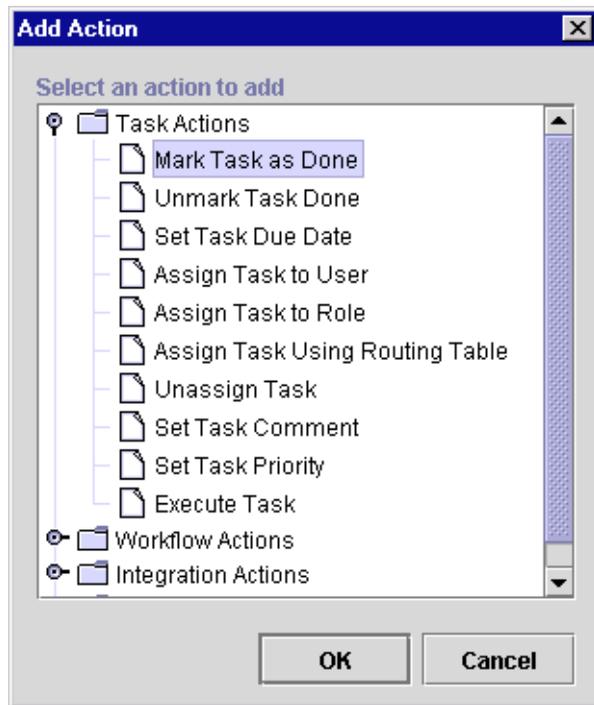
8. From the Assign result to drop-down list, select the variable Inventory, and click OK.

You have now placed the value of the variable ItemNumber as the input parameter to the JavaBean POBean and the variable Inventory will get the result back from the execution of this JavaBean.

You will now add the Mark Task as Done action to the Activated tab of the Task Properties dialog box. This action will mark the task as done and instantiate the C2 Decision node. Make sure the Activated tab is selected.

9. Click the Add button to display the Add Action dialog box.
10. Under Task Actions, select the Mark Task as Done action, and click OK.

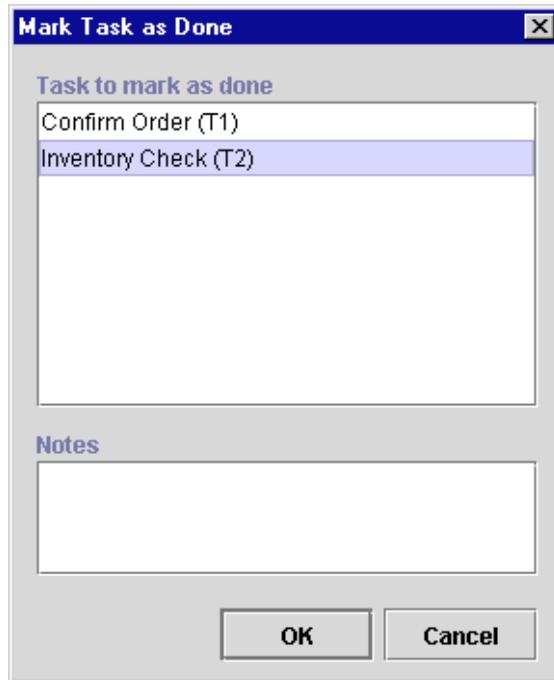
Figure 2-40 Add Action Dialog Box



The Mark Task as Done dialog box is displayed.

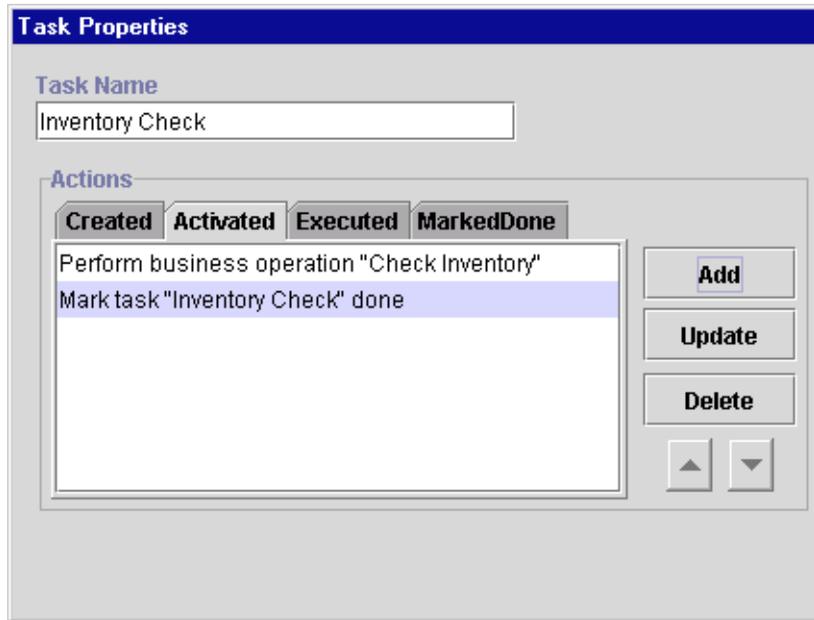
11. Select the Inventory Check (T2) task as the task to be marked done, and click OK.

Figure 2-41 Mark Task as Done Dialog Box



The Activated tab of the Task Properties dialog box now contains two tasks that will be executed upon activation of the workflow.

Figure 2-42 Activated Tab



Defining Decisions

A decision allows you to control the flow by evaluating a condition as true or false. Actions can be defined within the decision for each true or false case. The Order Processing workflow contains two decisions: one decision evaluates whether the user has confirmed the order, and the other confirms whether the inventory is available. If the inventory is available, the Order Processing workflow starts the ShipBill workflow, discussed in Chapter 4, “Defining the ShipBill Workflow.”

Decision: Has Order Been Confirmed?

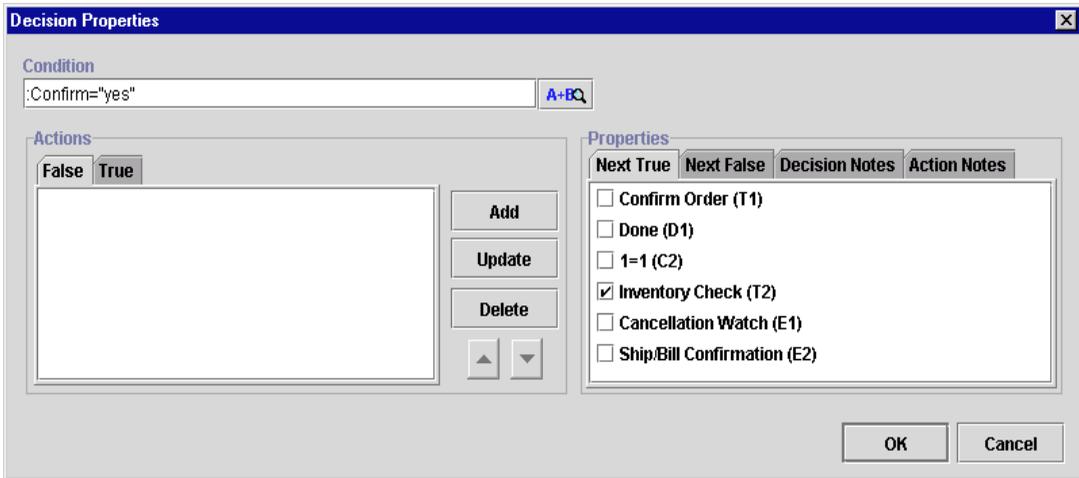
The first decision checks whether the user has confirmed the order. If the order has been confirmed, the flow continues to the next task Inventory Check. If the order has not been confirmed, an XML message is sent to cancel the workflow; the Cancellation Watch event in the workflow is triggered by this XML message, and the succeeding done node will cause the workflow to cancel.

Defining the C1 Decision Node

To define the C1 decision node:

1. With the Order Processing workflow diagram open in the drawing area, double-click the C1 decision shape to display the Decision Properties dialog box.

Figure 2-43 Decision Properties Dialog Box



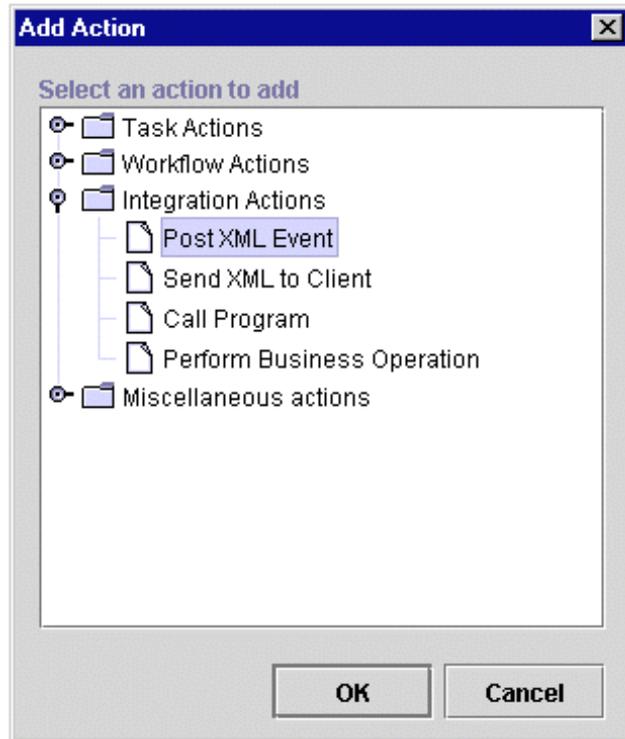
2. In the Condition field, enter `:Confirm="yes"` to rename the C1 decision node. Note that this is an expression.

Note: You can type this expression directly into the Condition description field or can construct it using the Expression Builder, which you can access by clicking the A+B button.

This condition evaluates whether the user has responded Yes or No to the confirmation of the order. If the condition is true (the user has double-clicked Confirm Order in the worklist and responded Yes to the Confirm Order message), the flow moves to the next node. If the condition is false, an XML message is sent to the workflow server, which cancels the order.

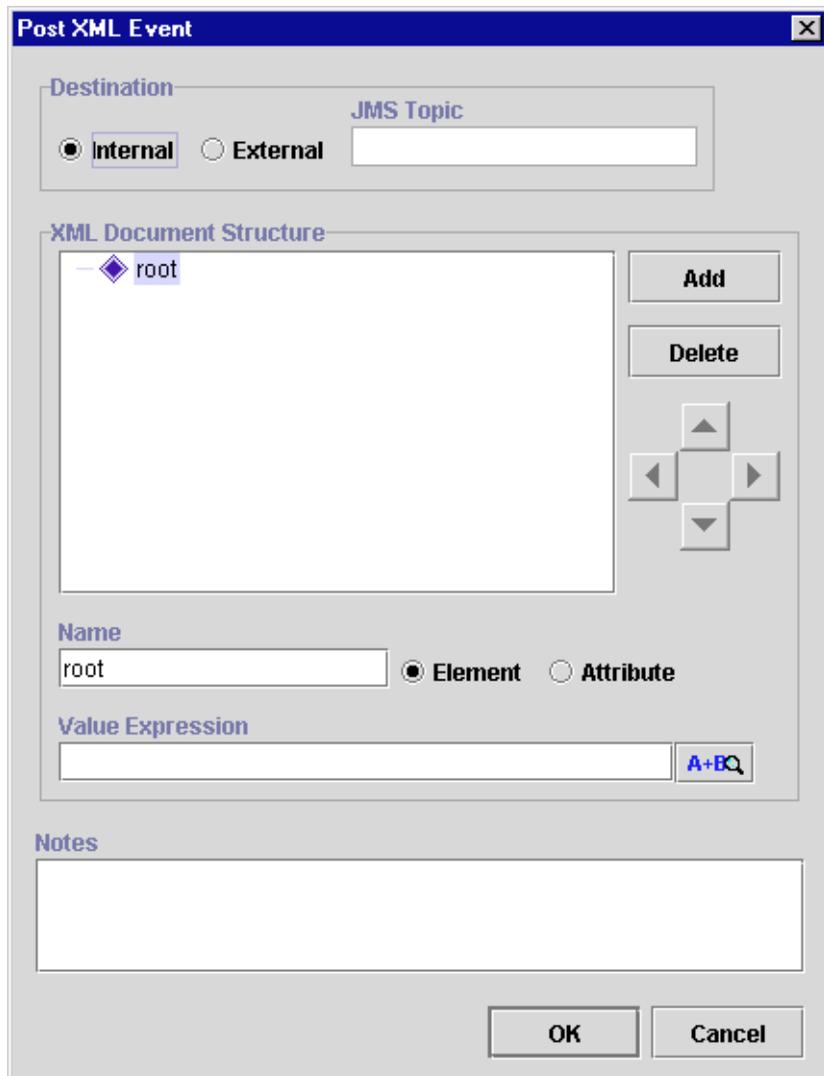
3. The action that allows you to define an XML document and post it is Post XML Event in the Integration Actions category. Select the False tab in the Decision Properties dialog box. In the False tab, you will set up the Post XML Event action. If the condition evaluates to false (if the order is not confirmed by the user joe), the event Cancellation Watch will be triggered by this particular XML document. You will set up the Cancellation Watch event in “Defining Events.”
4. Click Add to display the Add Action dialog box.

Figure 2-44 Decision Properties Dialog Box



5. Under Integration Actions, select Post XML Event and click OK to display the Post XML Event dialog box.

Figure 2-45 Post XML Event Dialog Box



Defining the XML Document Structure

In the Post XML Event dialog box, you define an XML document. In this example, the XML document has the root element `order` and two sub-elements:

- `status` having a value of "cancelled"
- `itemnumber` identifying the order that is being cancelled

To define the XML document structure:

1. Select the root element `root`, and in the Name field, type `order`.
2. Select the Element radio button.
3. Click Add to insert a new XML node in the XML document structure.
4. Highlight the new XML node, and in the Name field, type `status`.
5. Select the Element radio button and click OK.
6. In the Value Expression field, type "cancelled".
7. Click Add to insert a new XML node in the XML document structure.
8. Highlight the new XML node, and in the Name field, enter `itemnumber`.
9. Select the Element radio button, and click OK.
10. In the Value expression field, enter `:ItemNumber`.

Figure 2-46 Post XML Event Dialog Box

Post XML Event

Destination

Internal External

JMS Topic

XML Document Structure

order

- status
- itemnumber

Add

Delete

Name

itemnumber Element Attribute

Value Expression

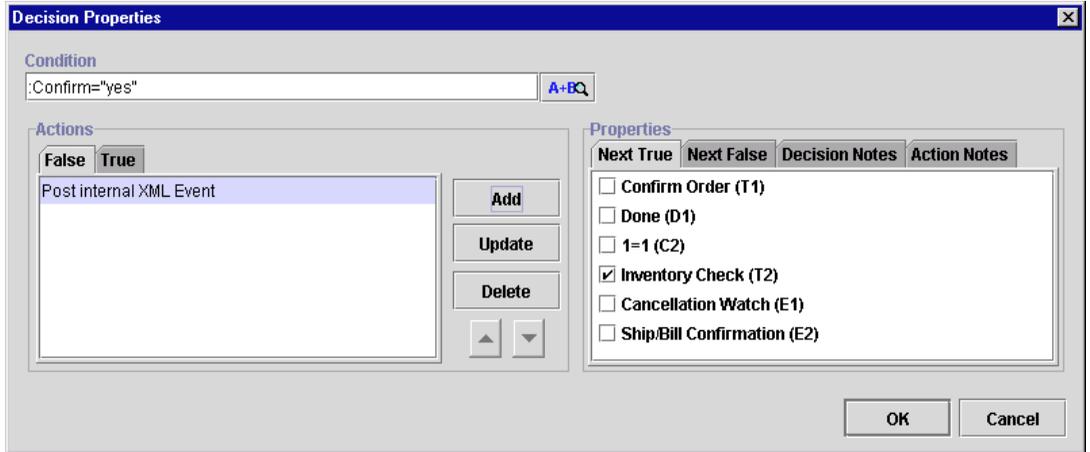
:itemNumber **A+B**

Notes

OK **Cancel**

11. Click OK to add the action to the False tab of the Decision Properties dialog box.

Figure 2-47 Decision Properties Dialog Box



As you will see in the “Defining Events” section, the event Cancellation Watch will be triggered by this particular XML document.

Decision: Is Item Available?

This decision evaluates the value of a variable that is set as the result of a call to the JavaBean POBean. The value is the quantity in stock of the ordered item. If this number is zero, the order is cancelled. If the number is greater than zero, the ShipBill workflow is started by the Start Workflow action and the flow moves on to the next node, which is the Ship/Bill Confirmation event. (In this example, the available number of items is greater than the numbers of items ordered.)

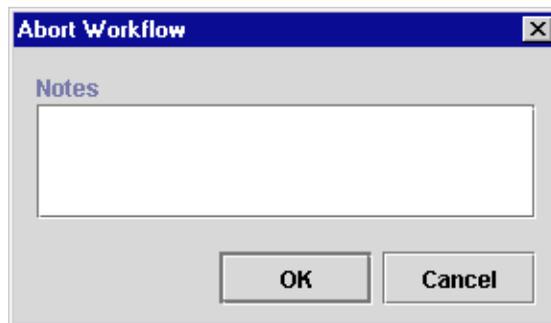
To define the C2 Decision node:

1. In the workflow diagram, double-click the C2 Decision shape to open the Decision Properties dialog box.
2. In the Condition field, type `:Inventory<>0`. (This means “Inventory not equal to zero.”)

This expression can be typed directly into the Condition description field or can be constructed using the Expression Builder, accessed by clicking the A+B button. This condition evaluates whether the user has responded Yes or No to the confirmation of the order.

3. Select the False tab and Click Add.
4. In the Add Action dialog box under Workflow Actions, select the Abort Workflow action and click OK to display the Abort Workflow dialog box.

Figure 2-48 Abort Workflow Dialog Box

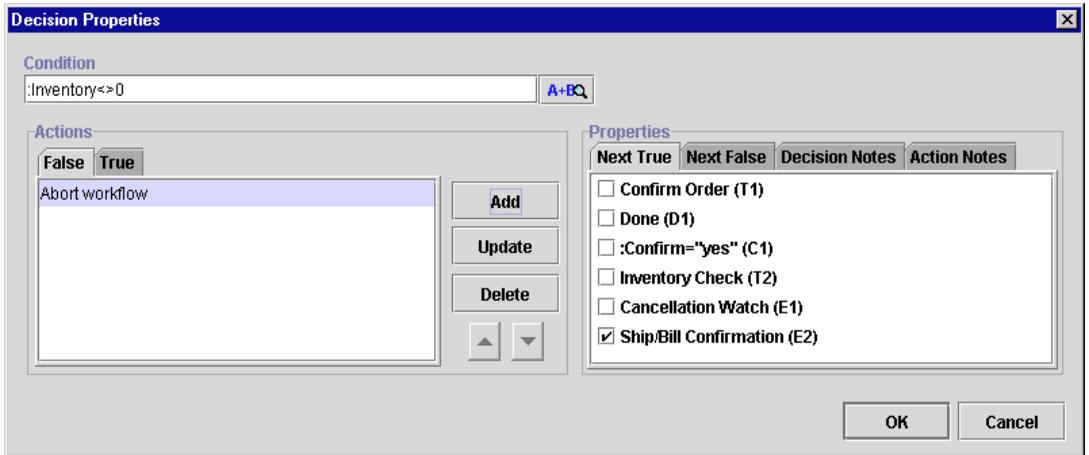


5. You do not need to enter anything in this dialog box. Simply click OK.

2 Defining the Order Processing Workflow

- The Abort Workflow action is added to the False tab in the Decision Properties dialog box. If the condition “Inventory not equal to zero” (:Inventory<>0) is false, the Abort Workflow action causes the workflow to cancel and the order cannot be completed.

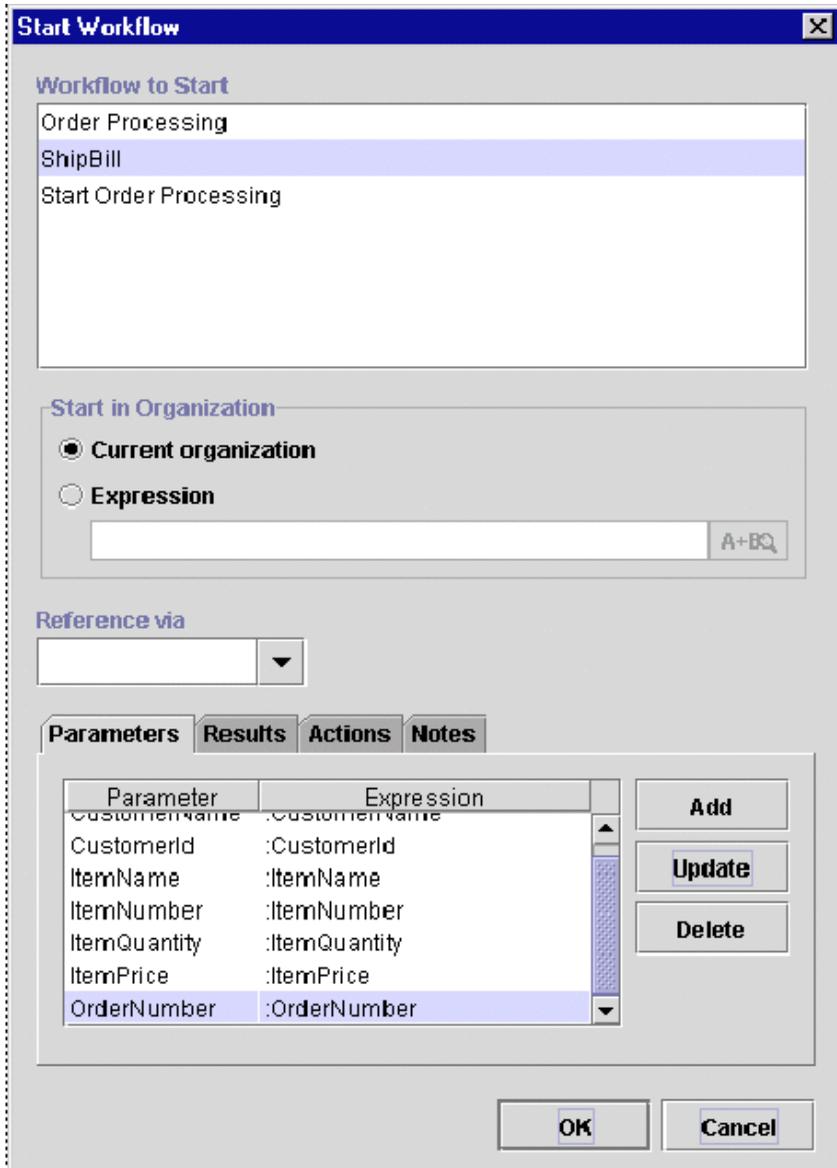
Figure 2-49 Decision Properties Dialog Box: Abort Workflow Action



- Select the True tab and click Add to display the Add Action dialog box.
- Under Workflow actions, select the Start Workflow action and click OK to display the Start Workflow dialog box.

The Start Workflow action enables you to start a sub-workflow (in this case, ShipBill) from the current workflow, Order Processing.

Figure 2-50 Start Workflow Dialog Box



9. In the Workflow to Start portion of the dialog, select ShipBill.
10. Under Start in Organization, select Current Organization.

In this example, the organization will remain ORG1, and the value of some variables are transferred from the Order Processing workflow to the ShipBill workflow.

As the example progresses, the variables in the “child” workflow (ShipBill), which will receive values from the “parent” workflow (Order Processing), will be defined as input variables within the variable definition dialog box. All input variables of the child workflow will automatically be displayed. Ordinarily, you can then use the Expression Builder to assign values to them. (In the example, we are simply transferring the values of the variables in the current workflow.)

Defining Events

An event is a notification node. An event is triggered by the arrival of an XML document that is produced internally within the WebLogic Process Integrator processing engine or from an external source through a Java Message Service (JMS) topic. When the flow reaches an event node, it waits until that event is triggered. An event can be cancelled by using the Cancel XML Event action.

The example contains two events: one event waits for the cancellation message, and the other event is triggered upon receipt of the shipping and billing confirmation in order to complete the order.

Event: Cancellation Watch

To define the Cancellation Watch event:

1. In the Order Processing workflow diagram, double-click the Cancellation Watch (E1) event to open the Event Properties dialog box.

Figure 2-51 Event Properties Dialog Box

The image shows a screenshot of the 'Event Properties' dialog box for a 'Cancellation Watch' event. The dialog has a title bar with a close button. It contains several input fields and buttons:

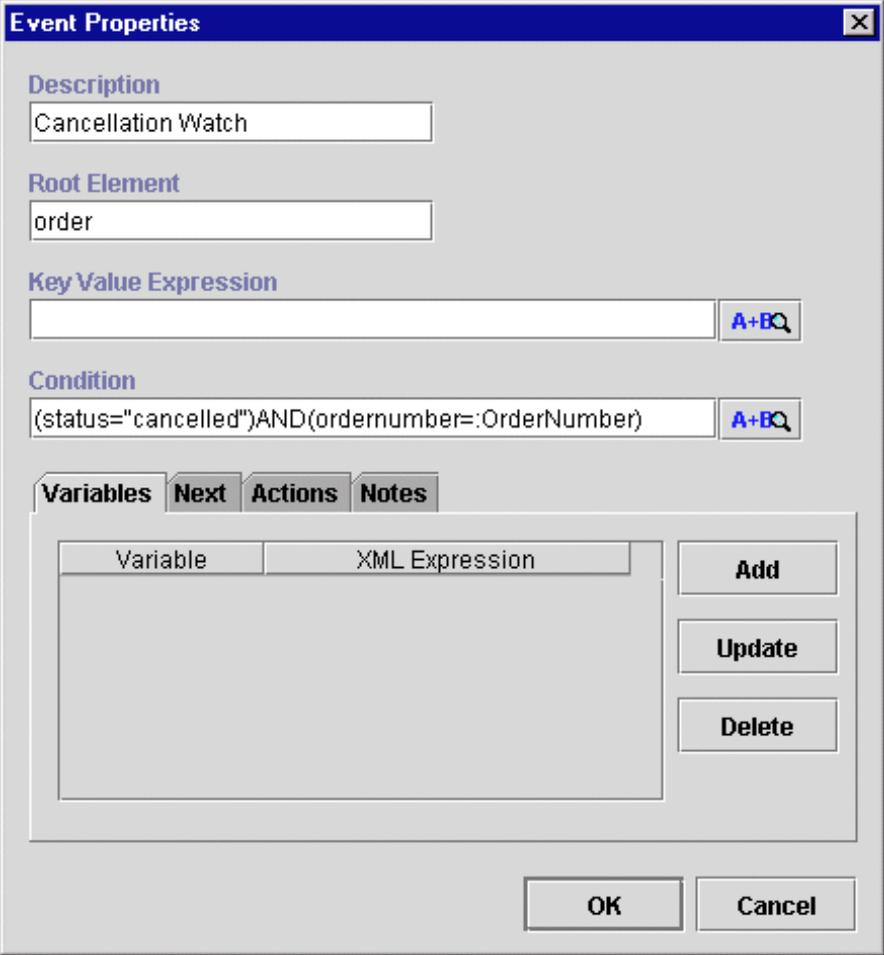
- Description:** A text box containing 'Cancellation Watch'.
- Root Element:** An empty text box.
- Key Value Expression:** An empty text box with an 'A+BQ' button to its right.
- Condition:** An empty text box with an 'A+BQ' button to its right.
- Variables:** A tabbed section with 'Variables', 'Next', 'Actions', and 'Notes' tabs. The 'Variables' tab is active, showing a table with two columns: 'Variable' and 'XML Expression'. The table is currently empty. To the right of the table are three buttons: 'Add', 'Update', and 'Delete'.
- OK and Cancel:** Two buttons at the bottom right of the dialog.

2 Defining the Order Processing Workflow

2. In the Root Element field, enter the root element of the XML document: `order`. As described in the “Decision: Has Order Been Confirmed?” section, `order` is the root element of the XML message sent by the Confirm Order task if the user does not confirm the order.
3. In the Condition field, enter the condition to be satisfied. In this case, the condition is for the status to be “cancelled” and the ID of the workflow to match the value of the variable `OrderNumber`. Enter the condition as an expression in the Condition field or use the Expression Builder to compose the expression by clicking the A+B button. The expression in the Condition field should be:

```
(status="cancelled")AND(ordernumber=:OrderNumber)
```

Figure 2-52 Event Properties Dialog Box



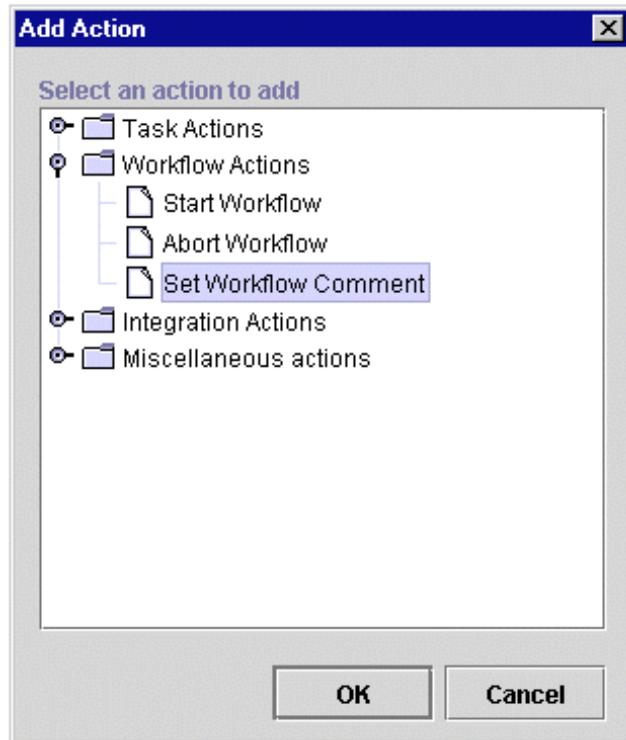
The dialog box is titled "Event Properties" and contains the following fields and controls:

- Description:** A text box containing "Cancellation Watch".
- Root Element:** A text box containing "order".
- Key Value Expression:** An empty text box with an "A+BQ" button to its right.
- Condition:** A text box containing "(status="cancelled")AND(ordernumber=:OrderNumber)" with an "A+BQ" button to its right.
- Variables:** A tabbed section with "Variables", "Next", "Actions", and "Notes" tabs. The "Variables" tab is active, showing a table with two columns: "Variable" and "XML Expression". The table is currently empty. To the right of the table are three buttons: "Add", "Update", and "Delete".
- OK** and **Cancel** buttons are located at the bottom right of the dialog.

No further actions will be performed within this event, and the flow will move to the next node, which is the done node. However, in order to distinguish between a cancelled order and a completed one (having reached the done node of the workflow), you will add a workflow comment that will be displayed on the monitoring window upon execution of this workflow and when the cancellation occurs.

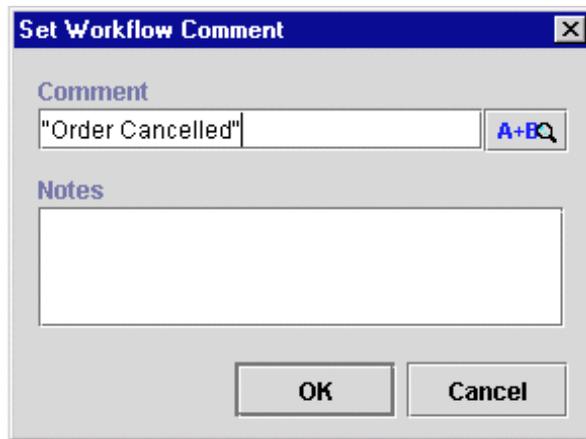
4. Select the Actions tab and click Add to display the Add Action dialog box.

Figure 2-53 Add Action Dialog Box



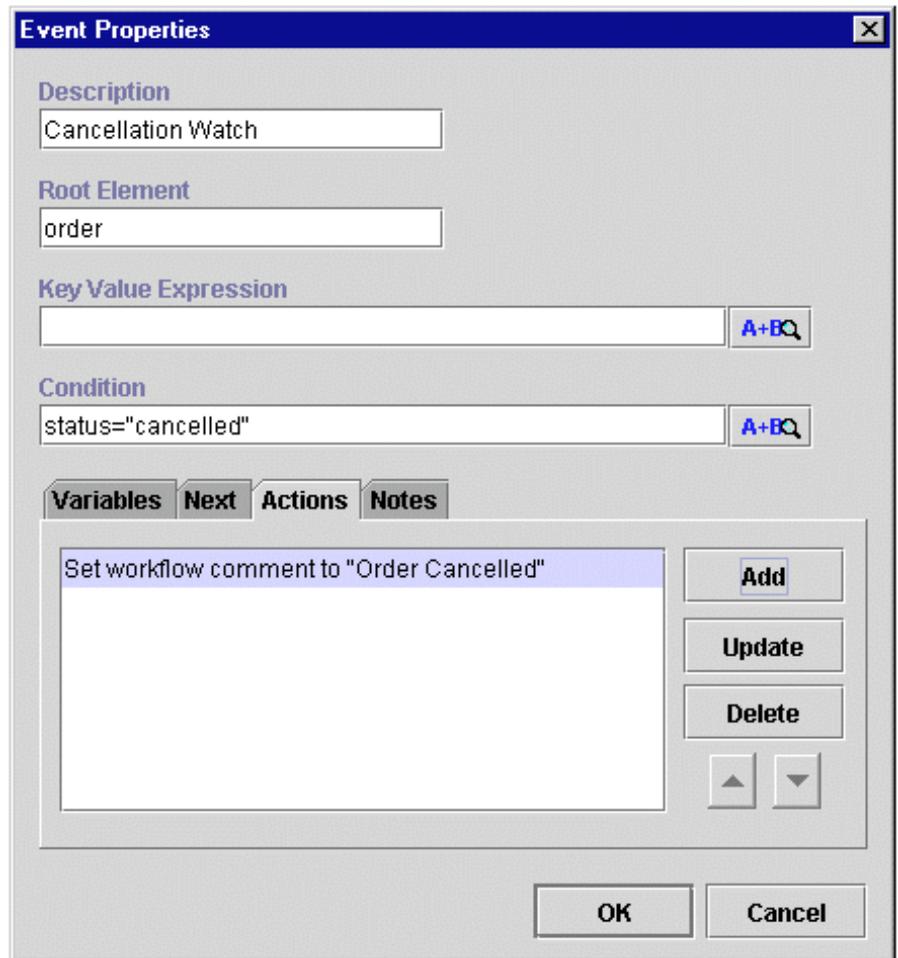
5. In the Add Action dialog box under Workflow Actions, select Set Workflow Comment and click OK.
6. In the Set Workflow Comment dialog box, type "Order Cancelled".

Figure 2-54 Set Workflow Comment Dialog Box



7. Click OK to add the action to the Actions tab of the Event Properties dialog box.

Figure 2-55 Event Properties Dialog Box



Event: ShipBill Confirmation

Next, you set up the ShipBill Confirmation event, which waits for the ShipBill workflow to send back an XML document confirming the shipping and billing for the ordered item. (The ShipBill workflow is predefined for this example and is discussed in Chapter 4, “Defining the ShipBill Workflow.”)

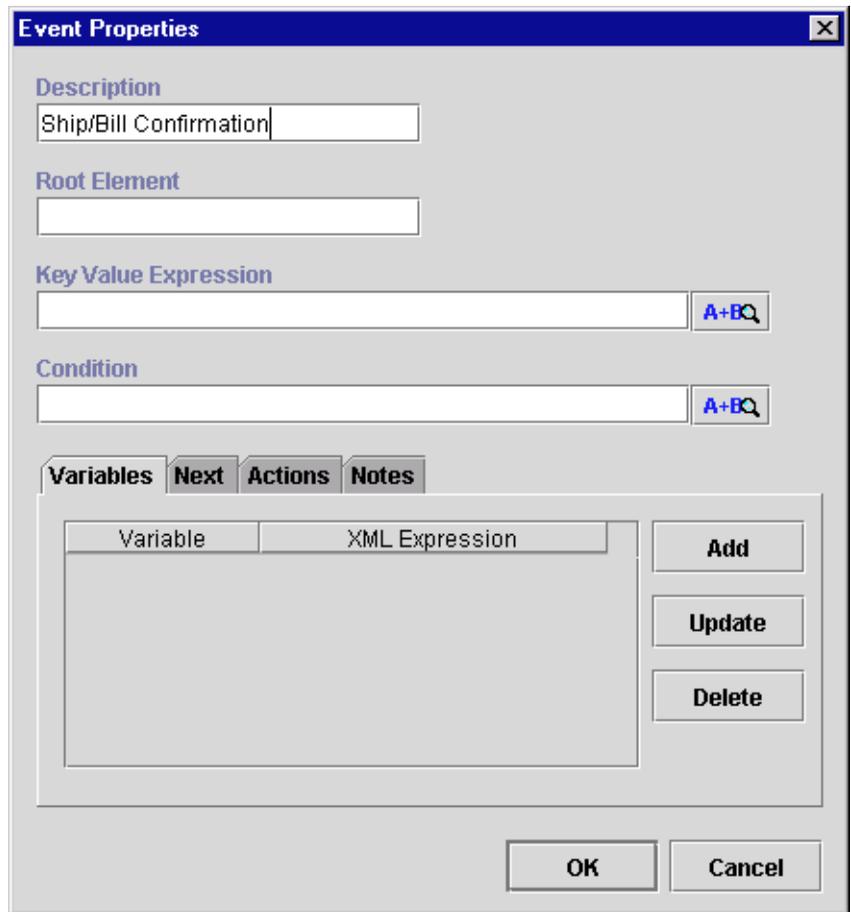
The root element of the XML message that will trigger this event is `order`, the same element that triggers the Cancellation Watch event. The condition, however, is different. For this event to be triggered, the status has to be “complete”.

The total price of the item calculated by the JavaBean POBean in the ShipBill workflow is also returned to this workflow through the XML document.

To define the ShipBill Confirmation event:

1. In the Order Processing workflow diagram, double-click the ShipBill Confirmation (E2) event to open the Event Properties dialog box.

Figure 2-56 Event Properties Dialog Box



2. In the Root Element field, enter the root element of the XML document: `order`. As described in the “Decision: Has Order Been Confirmed?” section, `order` is the root element of the XML message sent by the Confirm Order task if the user does not confirm the order.

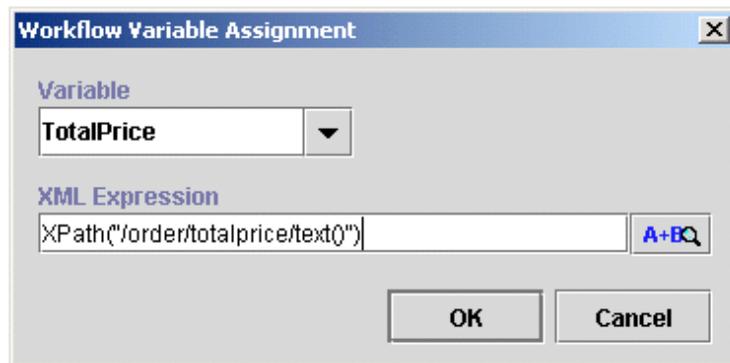
- In the Condition field enter the condition to be satisfied. In this case, the condition is for the status to be "complete". The status must be such in order to trigger the Ship/Bill Confirmation event. Enter the condition as an expression in the Condition field or use the Expression Builder to compose the expression by clicking the A+B button. The expression should read as follows:

```
status="complete"
```

Next, you will set up an XML document, which returns the total price of the item calculated by the JavaBean POBean in the ShipBill workflow to the Order Processing workflow.

- In the Event Properties dialog box, click the Add button to display the Workflow Variable Assignment dialog box.

Figure 2-57 Workflow Variable Assignment Dialog Box



- From the Variable drop-down list, select the variable TotalPrice.
- In the XML Expression field, enter the XML expression directly in the field or use the A+B button to access the Expression Builder to compose the expression, then click OK. The expression should read as follows:

```
XPath( "/order/totalprice/text()" )
```

Your final Event Properties dialog box should look like the following figure:

Figure 2-58 Final Event Properties Dialog Box

Event Properties

Description
Ship/Bill Confirmation

Root Element
order

Key Value Expression
A+BQ

Condition
status="complete" A+BQ

Variables | Next | Actions | Notes

Variable	XML Expression
TotalPrice	XPath("/order/totalprice/text()")

Add
Update
Delete

OK Cancel

Setting Up the Trigger

The Order Processing workflow is an event-triggered workflow. An XML document will be constructed and sent from the Start Order Processing workflow to trigger the Order Processing workflow. You specify the trigger conditions for the workflow in the Start node. There are four ways to start a workflow:

- Event—XML document
- Timed—Exact date and time start
- Manual—Manual start from the worklist
- Called—Called from another workflow by the Start Workflow action

This example uses the Event method. The root element of the XML document will be `order`, and the condition is `status="new"`. The organization for which this workflow should be instantiated also must be specified. Once this workflow is triggered, some of the variables will be populated by values contained in the XML document.

To set up the XML document that will trigger the Order Processing workflow:

1. In the Order Processing workflow diagram, double-click the Start node to display the Start Properties dialog box.
2. Select the Event radio button.

Figure 2-59 Start Properties Dialog Box: Event

Start Properties [X]

Description
Start

Event **Timed** **Manual** **Called**

Root Element
[]

Key Value Expression
[] **A+BQ**

Condition
[] **A+BQ**

Start Organization Expression
[] **A+BQ**

Variables | **Next** | **Actions** | **Notes**

Variable	XML Expression
----------	----------------

Add
Update
Delete

OK **Cancel**

3. In the Root Element field, enter the root element of the XML document: `order`. As described in the “Decision: Has Order Been Confirmed?” section, `order` is the root element of the XML message sent by the Confirm Order task if the user does not confirm the order.
4. In the Condition field, enter the condition to be satisfied. In this case, the condition is for the status to be “`new`”. The status must be such in order to trigger the Order Processing workflow whenever a new order is processed. Enter the condition as an expression in the Condition field or use the Expression Builder to compose the expression by clicking the A+B button. The expression should look like the following:

```
status="new"
```

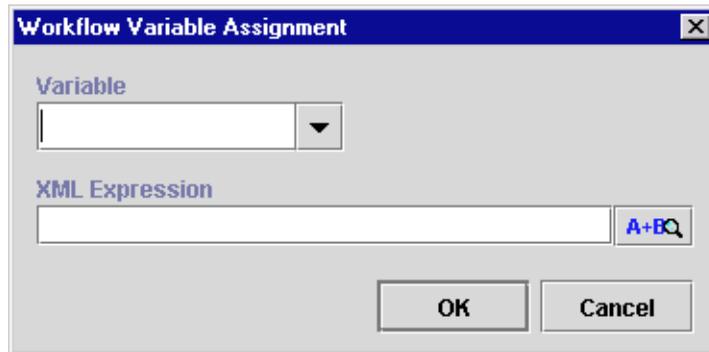
5. In the Start Organization Expression field, enter an expression indicating in which organization the Order Processing will be started:

```
"ORG1"
```

Next, you will construct an XML document by defining the variable `CustomerName` to extract the value of the element name from the root `customer`.

6. Click the Add button to display the Workflow Variable Assignment dialog box.

Figure 2-60 Workflow Variable Assignment Dialog Box



2 Defining the Order Processing Workflow

7. From the Variable drop-down list, select the variable `CustomerName`.
8. In the XML Expression field, enter the XML expression to extract the value of the element name from the root `customer`. Enter the XML expression directly in the field or use the A+B button to access the Expression Builder to compose the expression:

```
XPath("/order/customer/name/text()")
```

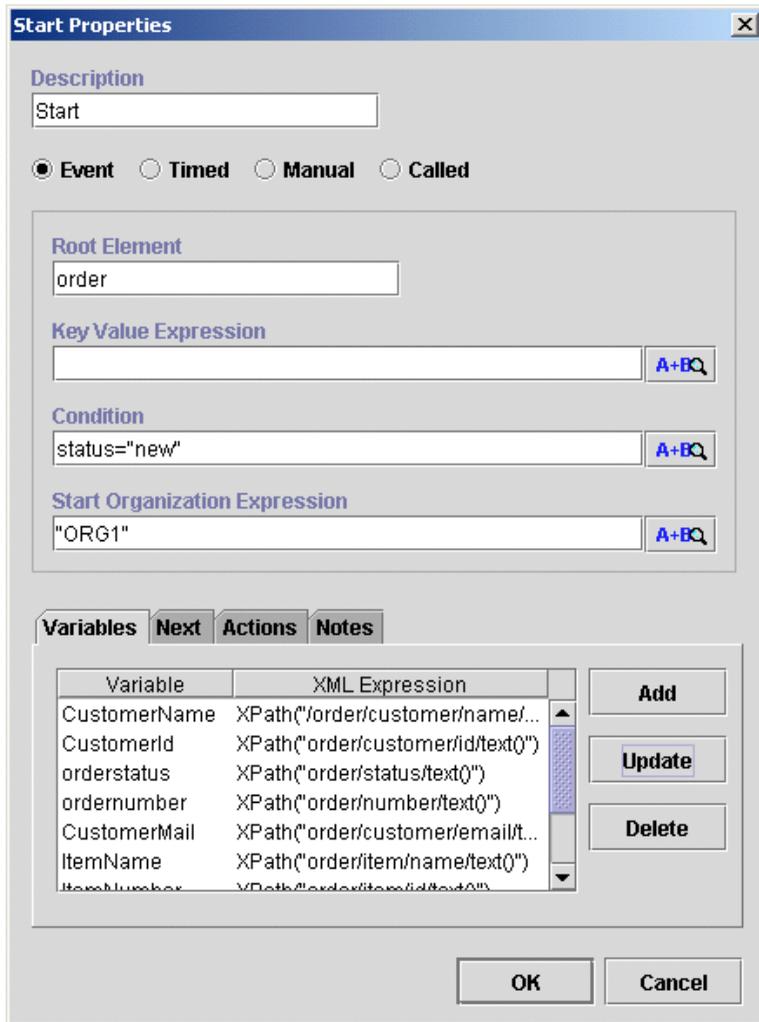
9. Click OK.

10. Repeat steps 7 through 9 for the following variables:

Table 2-3 Variable Expressions

Variable	Expression
CustomerId	<code>XPath("order/customer/id/text()")</code>
CustomerMail	<code>XPath("order/customer/email/text()")</code>
CustomerState	<code>XPath("order/customer/state/text()")</code>
ItemName	<code>XPath("order/item/name/text()")</code>
ItemNumber	<code>XPath("order/item/id/text()")</code>
ItemQuantity	<code>XPath("order/item/quantity/text()")</code>
ItemPrice	<code>XPath("order/item/price/text()")</code>
OrderStatus	<code>XPath("order/status/text()")</code>
OrderNumber	<code>XPath("order/number/text()")</code>

Figure 2-61 Start Properties Dialog Box



Setting Up the Workflow Template Definition Processing Properties

In order to process the Order Processing workflow, you must set up certain workflow template definition processing properties. The workflow must be marked as Active in the Properties dialog before it can be instantiated.

To set up the workflow template definition processing properties:

1. In the main window of the WebLogic Process Integrator Studio, right-click the Order Processing folder and choose Properties from the pop-up menu to display the Template Definition Order Processing dialog box.

Figure 2-62 Template Definition Order Processing Dialog Box

The dialog box is titled "Template Definition Order Processing". It features a search field for the "Id" with a magnifying glass icon and the text "A+BQ". The "Active" checkbox is checked. The "Effective" date is set to "Jul 12, 2000" and the "Expiry" date is set to "Jul 13, 2000". The "Enable auditing" checkbox is unchecked. The "Notes" field is empty. The "Last changed on" field shows "Jul 13, 2000 2:38:00 PM" and the "Last changed by" field shows "maryam". The "OK" and "Cancel" buttons are at the bottom right.

2. In the Id field, type "Order" + :Order Number.

The ID for the Order Processing workflow will be displayed on the worklist and on the monitoring screens. This ID is unique for each instance of this workflow; it is an expression built using workflow variables.

3. Select the Active checkbox to allow the instantiation of the Order Processing workflow at run time. (A workflow cannot start unless it is marked Active.)

You already defined the Effective and Expiry dates when you created the workflow template definition for Order Processing in "Creating a New Workflow Template Definition."

Figure 2-63 Template Definition Order Processing Dialog Box

The dialog box is titled "Template Definition Order Processing" and contains the following elements:

- Id:** A text field containing the expression `"Order"+:OrderNumber` and a search icon labeled `A+BQ`.
- Active:** A checked checkbox.
- Effective:** A date picker showing `Jul 12, 2000`.
- Expiry:** An unchecked checkbox and a date picker showing `Jul 13, 2000`.
- Enable auditing:** An unchecked checkbox.
- Notes:** A large empty text area.
- Last changed on:** A date and time field showing `Jul 13, 2000 2:38:00 PM`.
- Last changed by:** A text field containing the name `maryam`.
- Buttons:** `OK` and `Cancel` buttons at the bottom right.

4. Click OK.

3 Defining the Start Order Processing Workflow

The Start Order Processing workflow contains only one task, which, upon execution, creates an XML document that in turn triggers the Order Processing workflow.

The following sections describe the process of creating and defining the Start Order Processing workflow:

- Creating the Start Order Processing Workflow Template
- Importing the Start Order Processing Workflow Template Definition
- Defining the Start Order Processing Workflow Template Definition

Creating the Start Order Processing Workflow Template

Create a new workflow template called Start Order Processing by following the procedure in “Creating a New Workflow Template” in Chapter 2, “Defining the Order Processing Workflow.” This workflow template will serve as the container into which you will import the Start Order Processing workflow template definition.

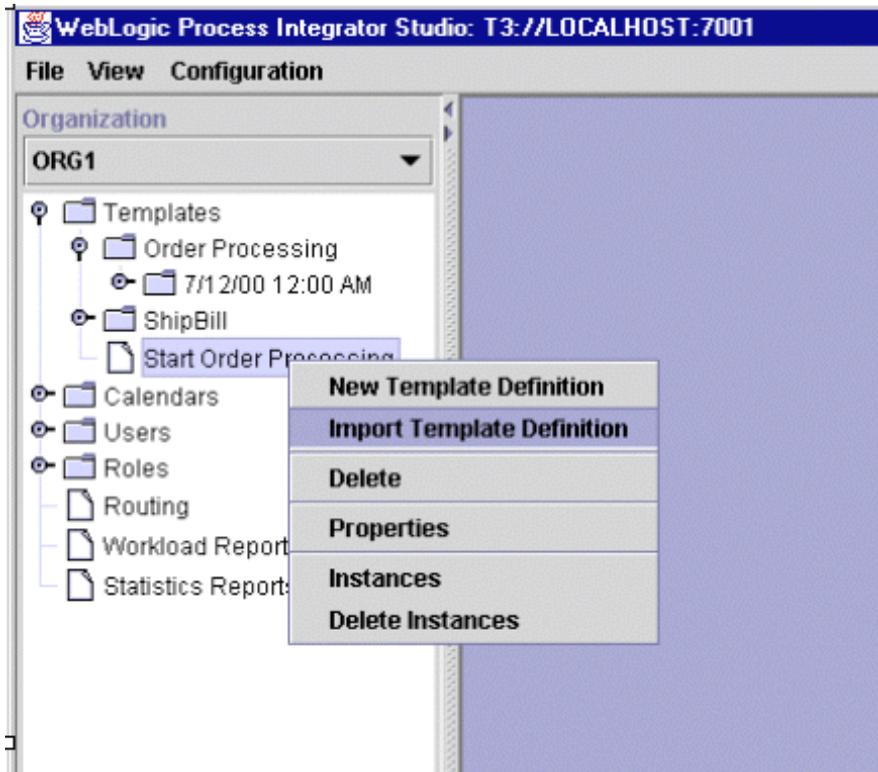
Importing the Start Order Processing Workflow Template Definition

You import the Start Order Processing workflow template definition from the `Examples` sub-directory found in the `Studio` directory, which you loaded onto your computer from the BEA WebLogic Process Integrator CD. All workflow template definitions can be exported and imported as XML documents.

To import the Start Order Processing workflow template definition:

1. In the folder tree of the main Studio window, right-click the Start Order Processing workflow template and choose Import Template Definition from the pop-up menu.

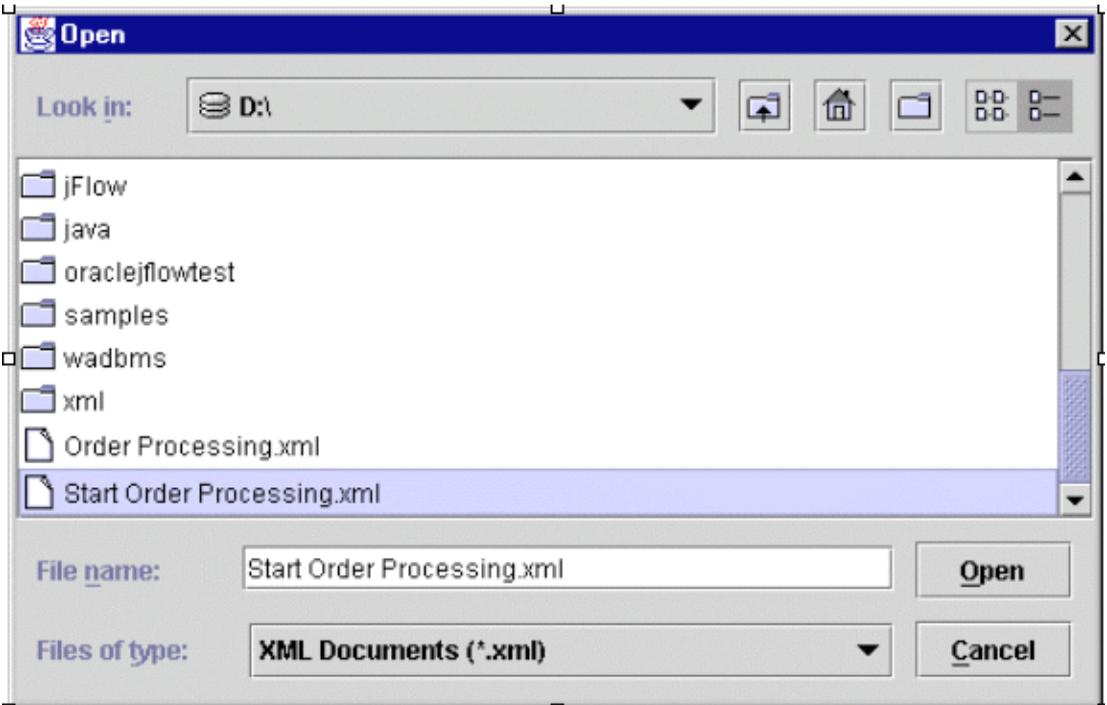
Figure 3-1 Select Import Template Definition



3 Defining the Start Order Processing Workflow

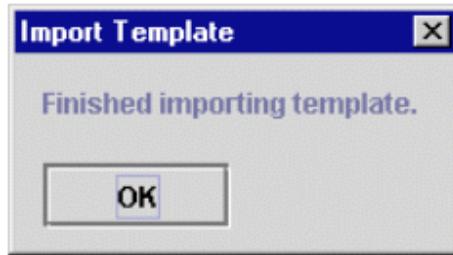
The Open dialog box is displayed.

Figure 3-2 Open Dialog Box



2. Select the Start Order Processing workflow from the directory where you installed the BEA WebLogic Process Integrator Studio product. The Start Order Processing workflow can be found in the `Examples` sub-directory of the `Studio` directory.
3. Click `Open`. A confirmation message is displayed when the workflow is successfully imported.

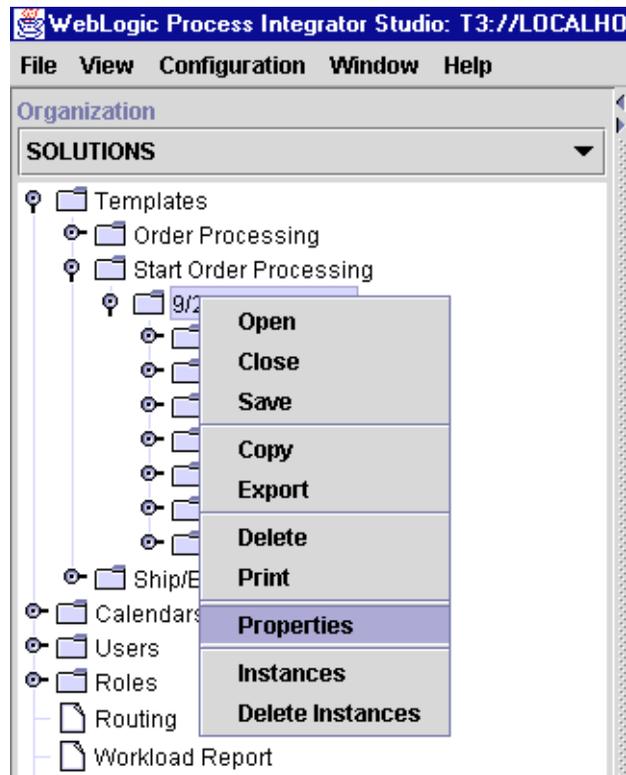
Figure 3-3 Import Template Dialog Box



An imported workflow is marked as inactive. You must mark it as Active before executing the workflow; an inactive workflow cannot be started.

4. In the WebLogic Process Integrator Studio main window, right-click the workflow template definition within the Start Order Processing workflow template.
5. From the pop-up menu, select Properties.

Figure 3-4 Select Properties from Menu



The Template Definition dialog box is displayed.

Figure 3-5 Template Definition Dialog Box

The screenshot shows a dialog box titled "Template Definition Start Order Processing" with a close button (X) in the top right corner. The dialog contains the following fields and controls:

- An "Id" text input field with an "A+BQ" button to its right.
- An unchecked checkbox labeled "Active".
- An "Effective" dropdown menu showing "Sep 20, 2000".
- An unchecked checkbox labeled "Expiry".
- An "Expiry" dropdown menu showing "Sep 20, 2000".
- An unchecked checkbox labeled "Enable auditing".
- A "Notes" text area.
- A "Last changed on" text field showing "Sep 20, 2000 4:19:00 PM".
- A "Last changed by" text field showing "student".
- "OK" and "Cancel" buttons at the bottom.

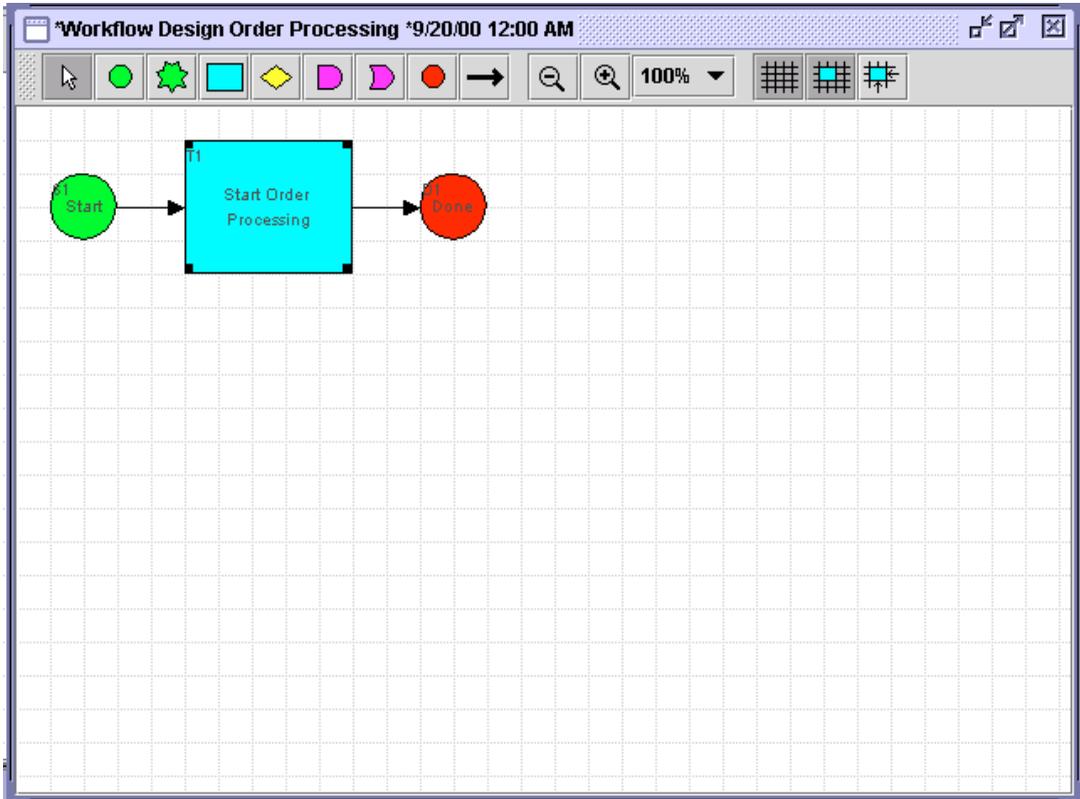
6. Select the Active dialog box to mark the imported Start Order Processing workflow template definition Active, and click OK.

Defining the Start Order Processing Workflow Template Definition

Because the workflow template definition for the Start Order Processing workflow was already defined before you imported it, you do not have to perform the tasks discussed in this section for the purpose of this tutorial. However, for your reference, screen shots of dialog boxes that already contain the setup information are provided, as well as an overview of what was involved in defining the Start Order Processing workflow.

For details on procedures for creating and setting up a workflow, see Chapter 2, “Defining the Order Processing Workflow.”

Figure 3-6 Start Order Processing Workflow

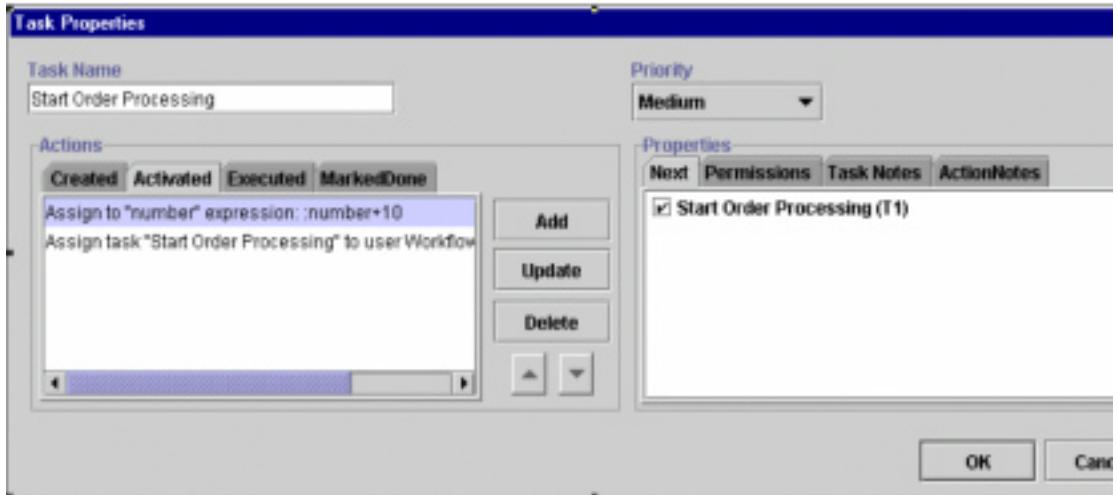


The Start node within the Start Order Processing workflow is set to a Manual start and must be started by a user from the Worklist application. Upon instantiation of this workflow, the first and only task, Start Order Processing, is displayed on the task list of the user who started the workflow. Upon the user's execution of the task, an XML document is created that triggers the Order Processing workflow.

In the workflow drawing area, double-click on the Start Order Processing task node to observe its task properties.

3 Defining the Start Order Processing Workflow

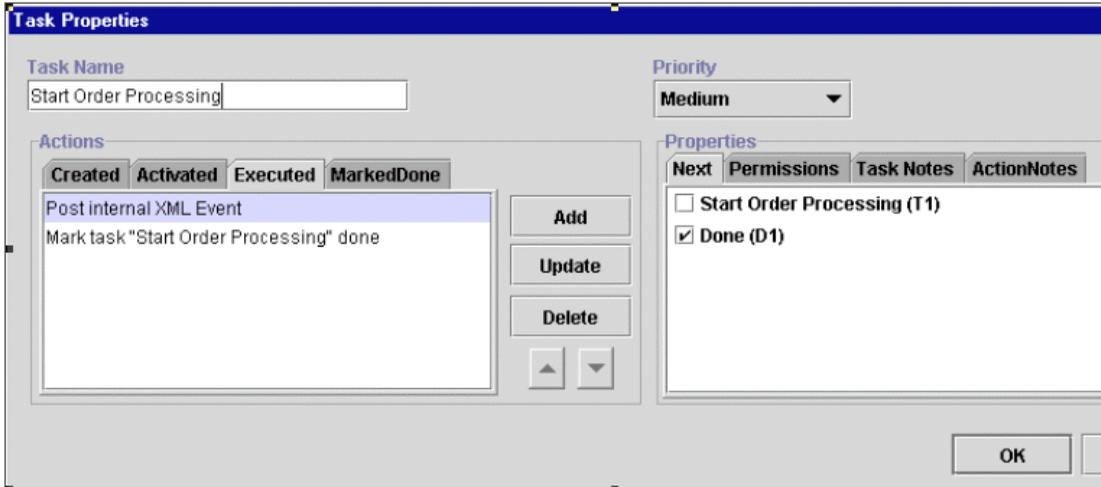
Figure 3-7 Task Properties: Start Order Processing



Under the Activated tab, the variable number is incremented and the task is assigned to the workflow initiator. Note that the successor to this task is itself. This workflow does not contain a done node and will actually never complete. Because we are using this workflow to start the entire ordering process, the recursive aspect of the Start Order Processing task allows you to start multiple instances of the Order Processing workflow by double-clicking the task in the drawing area (Figure 3-6).

Each time you double-click the task, the ID of the Order Processing workflow is incremented by 10, which produces a different instance. However, the details of the order contained in the XML document defined by the Post XML Event action will not change, and you will have to modify that action manually if you want to modify the details of the order.

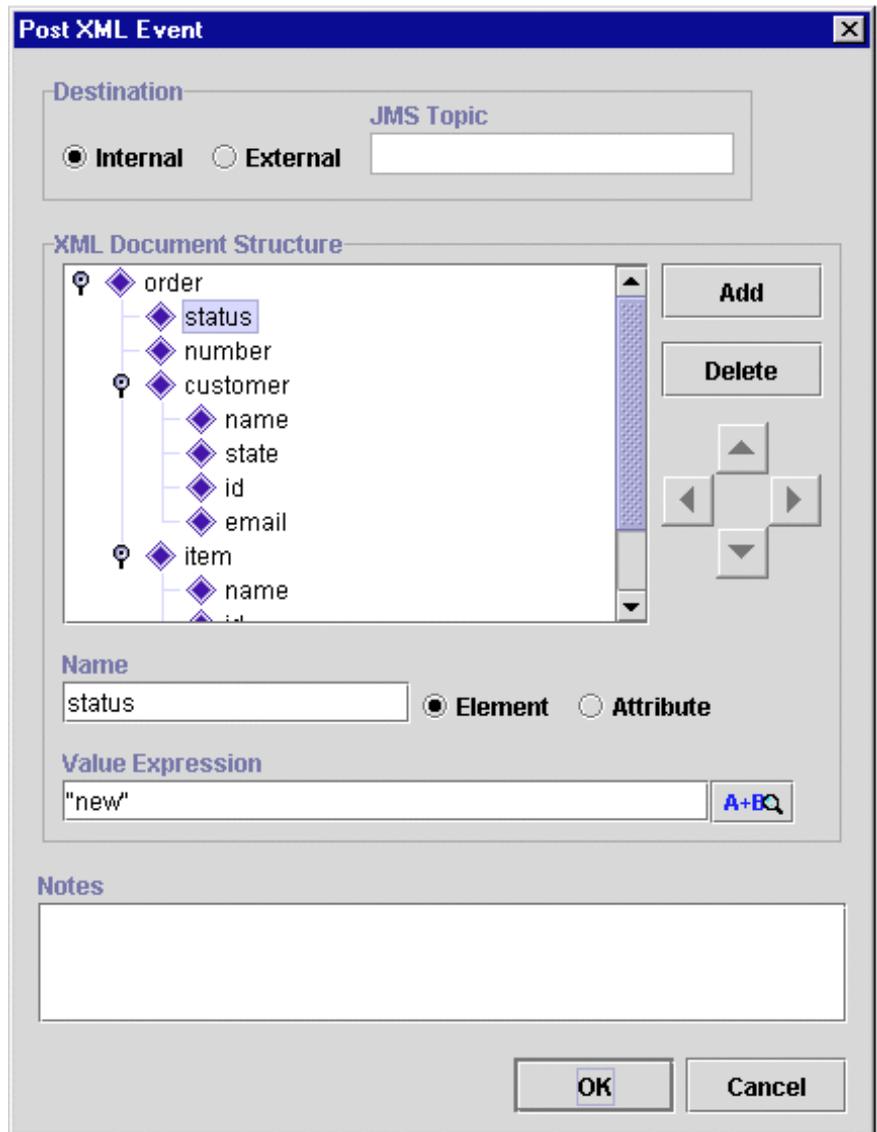
Figure 3-8 Task Properties: Post XML Event Action



Under the Executed tab, you define the Post XML Event action, which sends the XML document to start the Order Processing workflow. In the Post XML Event dialog box, you create the XML document structure. Refer to “Defining Events” in Chapter 2, “Defining the Order Processing Workflow.”

Select the Post Internal Event action and click Update to display the Post XML Event dialog box.

Figure 3-9 Post XML Event Dialog Box



The Post XML Event dialog box shows the following XML document:

Listing 3-1 XML Document

```
<Order>
  <status>new</status>
  <number>10</number>
  <customer>
    <name>John</name>
    <id>1234</id>
    <email>joe@workflow.ca</email>
    <state>CA</state>
  </customer>
  <item>
    <name>disk</name>
    <id>1314</id>
    <quantity >2</quantity >
    <price>24.5</price>
  </item>
</Order/>
```

Finally, you add a Mark Task as Done action to the Executed tab. This action completes this workflow because the successor to the task is the done node.

3 *Defining the Start Order Processing Workflow*

4 Defining the ShipBill Workflow

The ShipBill workflow handles the shipping and billing of the ordered item.

Import the ShipBill workflow template definition by using the procedure described in “Importing the Start Order Processing Workflow Template Definition” in Chapter 3, “Defining the Start Order Processing Workflow.”

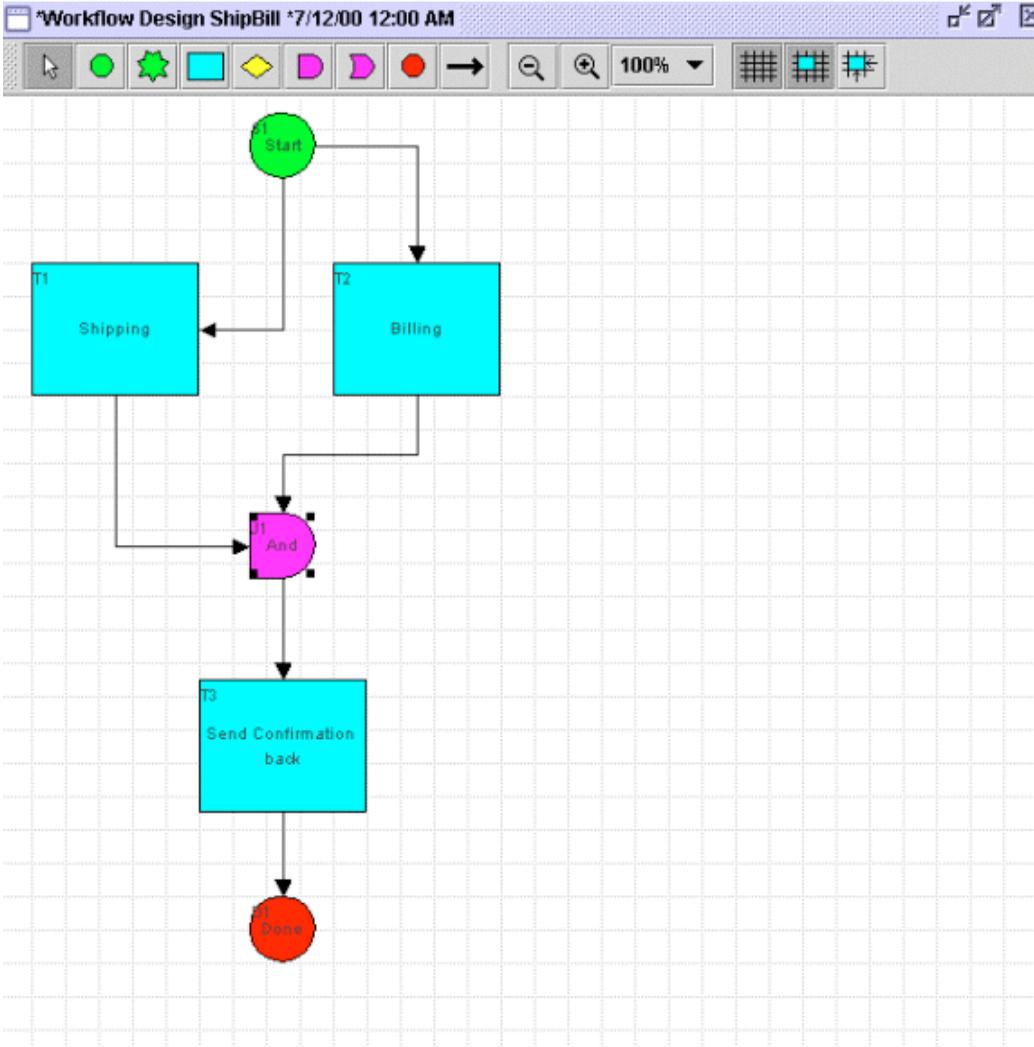
Once you import the workflow template definition for the ShipBill workflow, you do not have to perform the tasks discussed in this section for the purpose of this tutorial. However, for your reference, screen shots of dialog boxes that already contain the setup information are provided, as well as a brief overview explaining what was involved in defining the Ship/Bill workflow.

For details on procedures for creating and setting up a workflow, see Chapter 2, “Defining the Order Processing Workflow.”

4 Defining the ShipBill Workflow

Opening the ShipBill workflow template displays the following flow.

Figure 4-1 Drawing Area: ShipBill Workflow Diagram

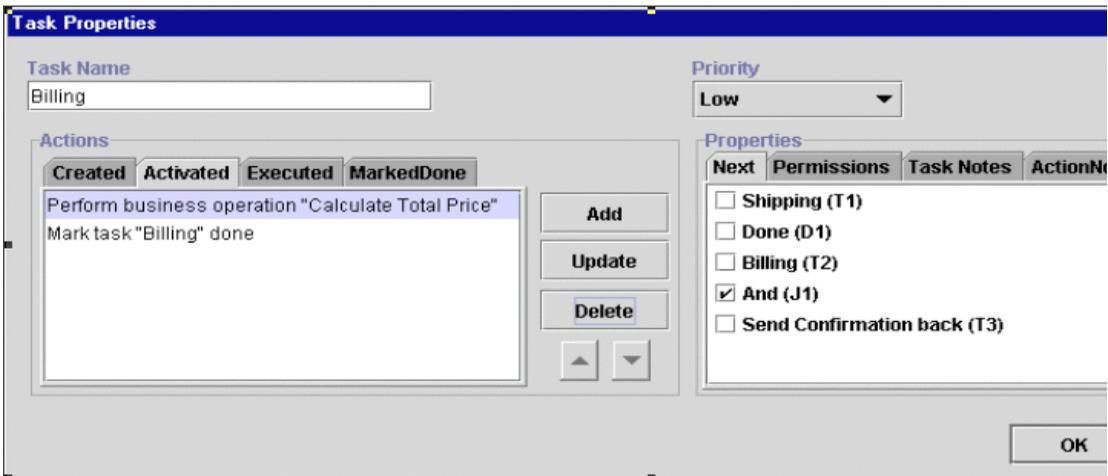


The ShipBill workflow is called by the Order Processing workflow, as defined in the C2 Decision node of the Order Processing workflow in “Defining Decisions” of Chapter 2, “Defining the Order Processing Workflow.” ShipBill is a sub-workflow to Order Processing. The value of the following variables are transferred from the Order Processing workflow to the ShipBill workflow:

- CustomerName
- CustomerId
- ItemName
- ItemNumber
- ItemQuantity
- ItemPrice
- OrderNumber

The Billing task calls the JavaBean POBean, which calculates the final price of the ordered item including tax and returns the value to the variable: TotalPrice.

Figure 4-2 Task Properties Dialog Box: Billing Task



The Perform Business Operation action is set up in the Activated tab as follows:

Figure 4-3 Perform Business Operation Dialog Box

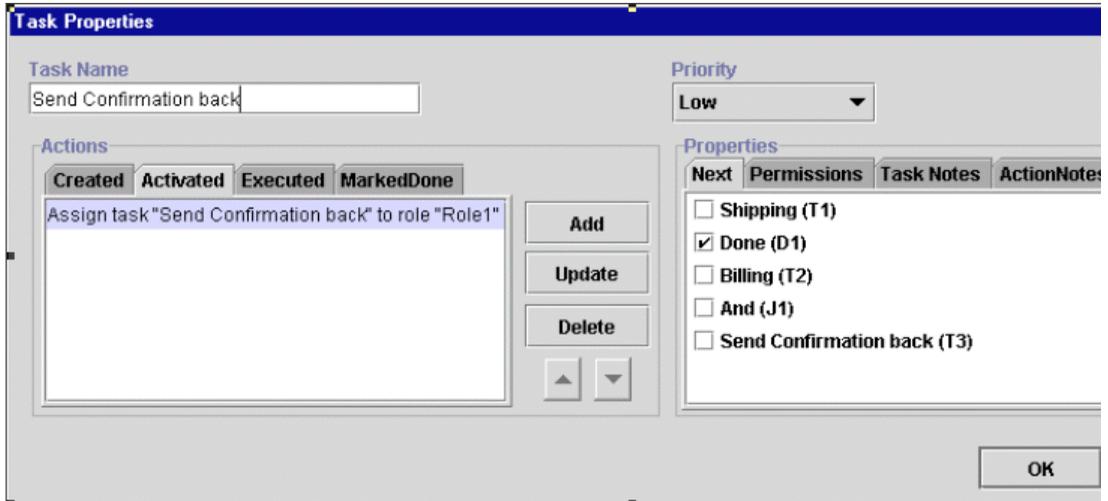
Name	Value
ItemNumber	:ItemNumber
ItemQuantity	:ItemQuantity
CustomerState	:CustomerState

The TotalPrice variable is then sent back to the Order Processing workflow by way of an XML message that is created and sent upon execution of the Send Confirmation Back task.

Note the use of the AND join in this workflow, whereby both tasks, Shipping and Billing, have to be marked as done before the flow moves on.

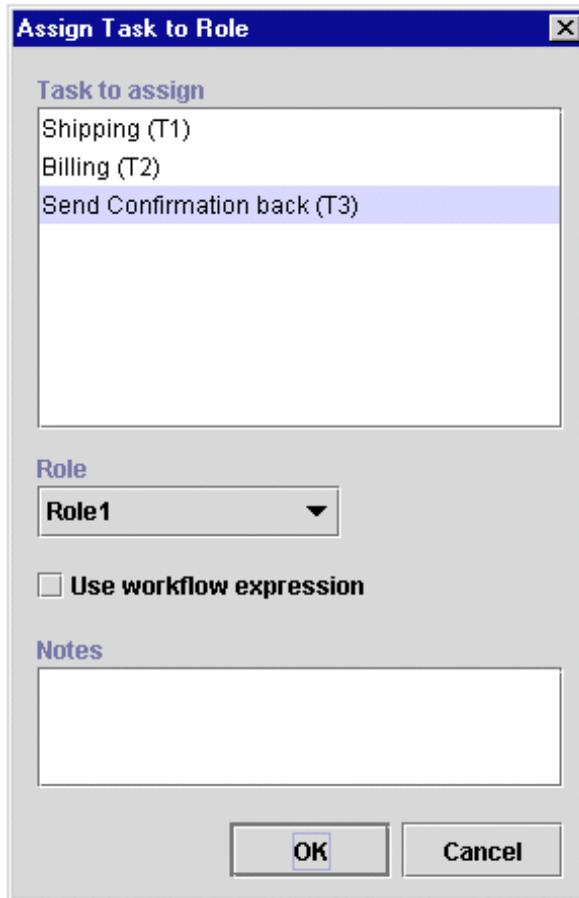
In the workflow diagram, double-click the Send Confirmation Back task to display the task properties.

Figure 4-4 Task Properties Dialog Box: Activated Tab



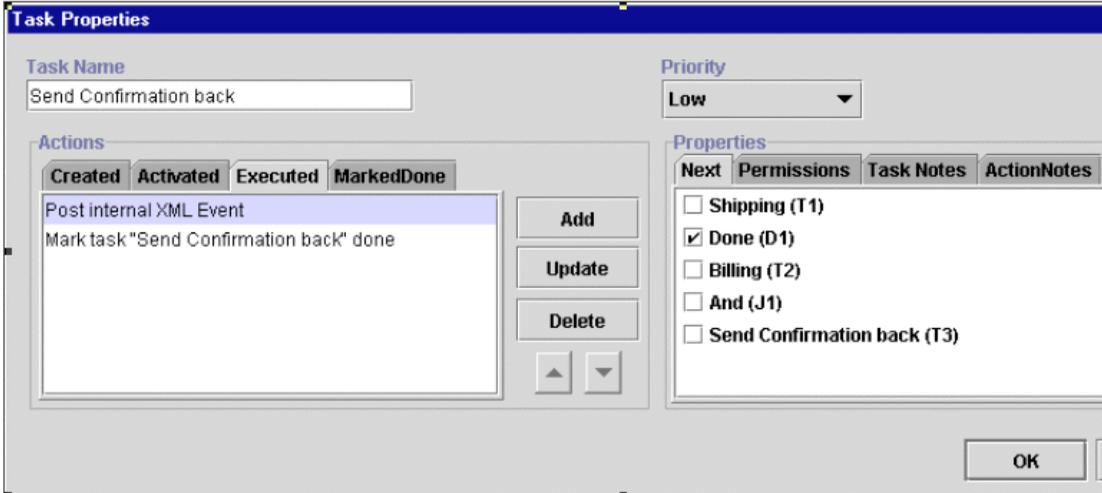
The Send Confirmation Back task is assigned to the role: Role1 by use of the Assign Task to Role action, as illustrated in the following dialog box.

Figure 4-5 Assign Task to Role Dialog Box



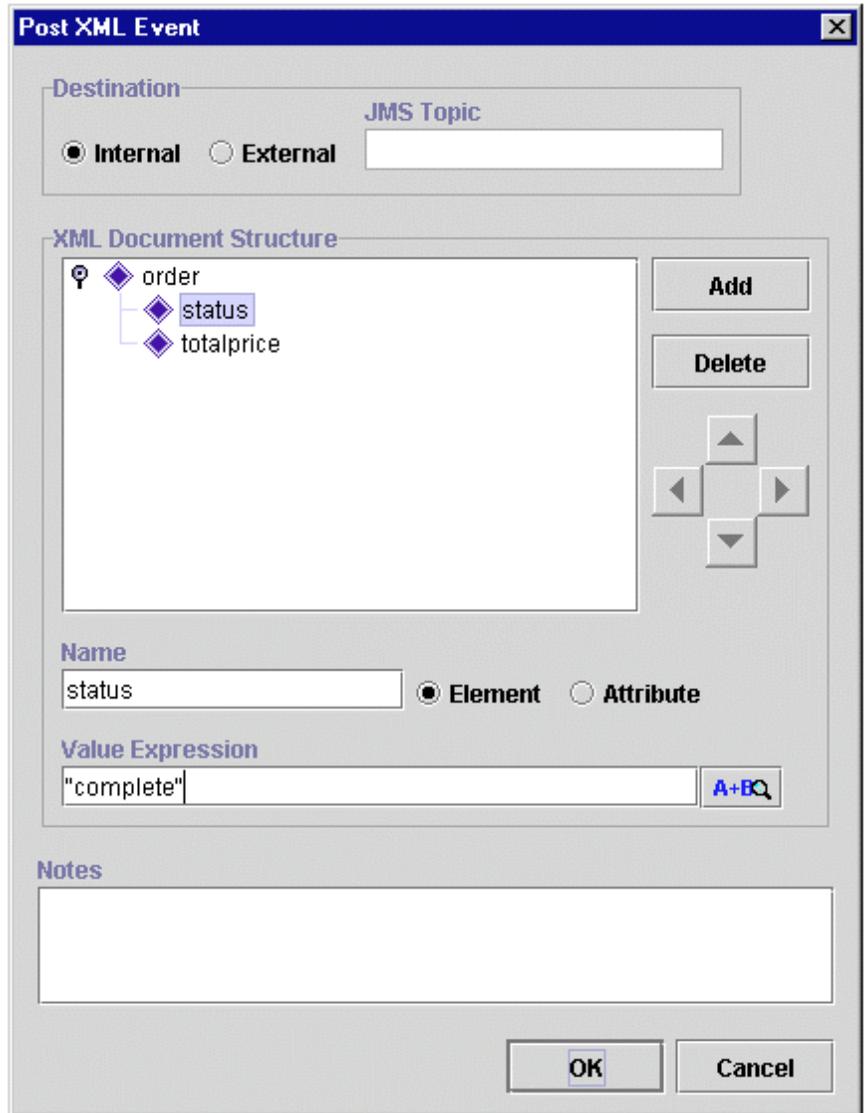
Select the Executed tab of the task.

Figure 4-6 Task Properties Dialog Box: Executed Tab



Upon execution of this task, an internal XML message is posted to the WebLogic Process Integrator server that will trigger the event in the Order Processing workflow and complete the entire process. Click Update on the Post Internal XML Event action.

Figure 4-7 Post XML Event Action Dialog Box: Internal XML Event



The XML document has the root element order and the status as complete. The value of the variable TotalPrice is also sent and will be read by the Order Processing workflow.

5 Executing the Workflow Example

The execution starting point for the workflow example is the Start Order Processing workflow. You start this workflow manually through the WebLogic Process Integrator Worklist application.

The following sections describe how to execute the workflow example:

- Logging On to the Worklist Application
- Executing the Workflow

Logging On to the Worklist Application

To start the worklist application:

1. From your desktop, choose Start > Programs > WebLogic Process Integrator > Worklist to display the Logon dialog box.

Figure 5-1 Logon to WebLogic Process Integrator Dialog Box



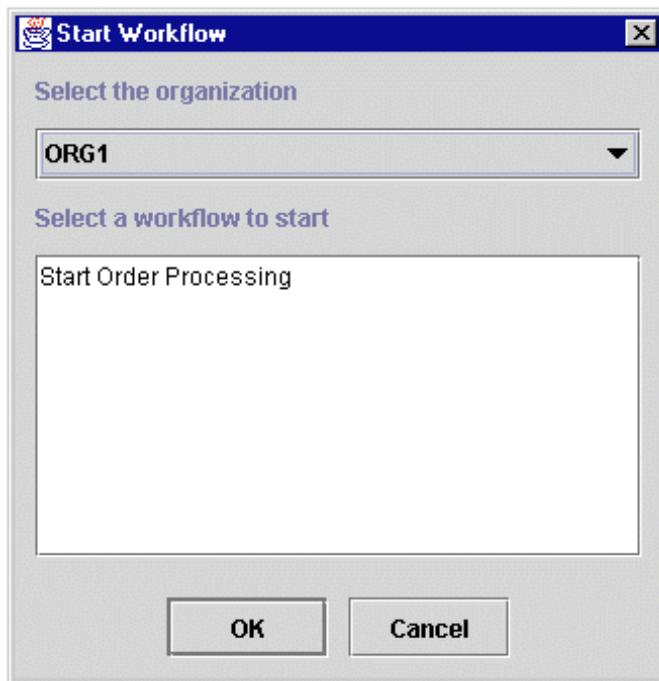
2. Enter your user ID, password, and the name of the WebLogic Process Integrator server to which you will connect.

Executing the Workflow

To execute the workflow example:

1. From the Workflow menu, choose Start Workflow to display the Start Workflow dialog box.

Figure 5-2 Start Workflow Dialog Box



2. Select the Start Order Processing workflow and click OK. The workflow is instantiated and a confirmation message is displayed.

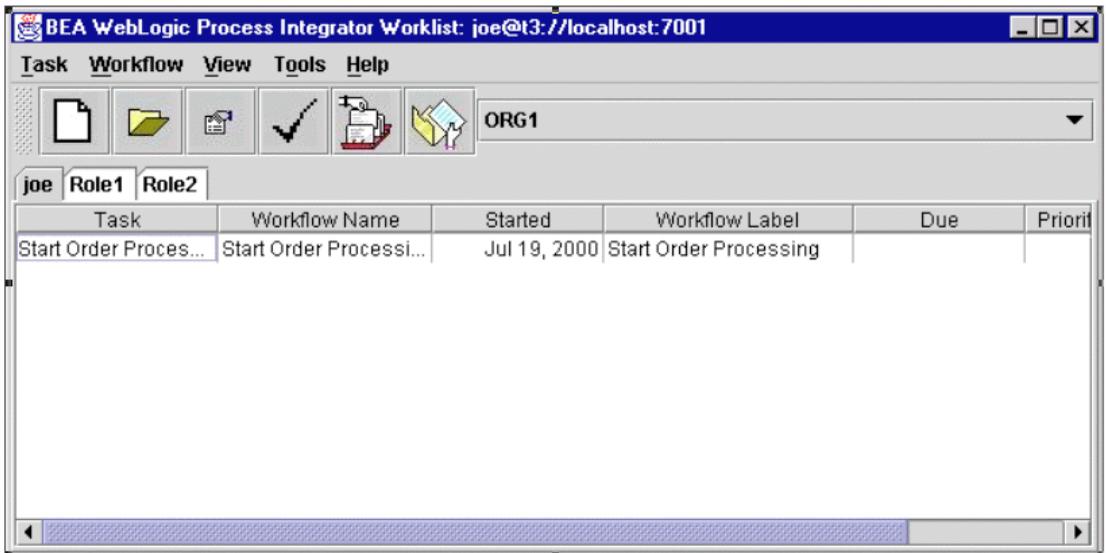
Figure 5-3 Workflow Successfully Started



3. Click OK.

The first and only task of the Start Order Processing workflow is assigned to the user joe and is displayed on the worklist.

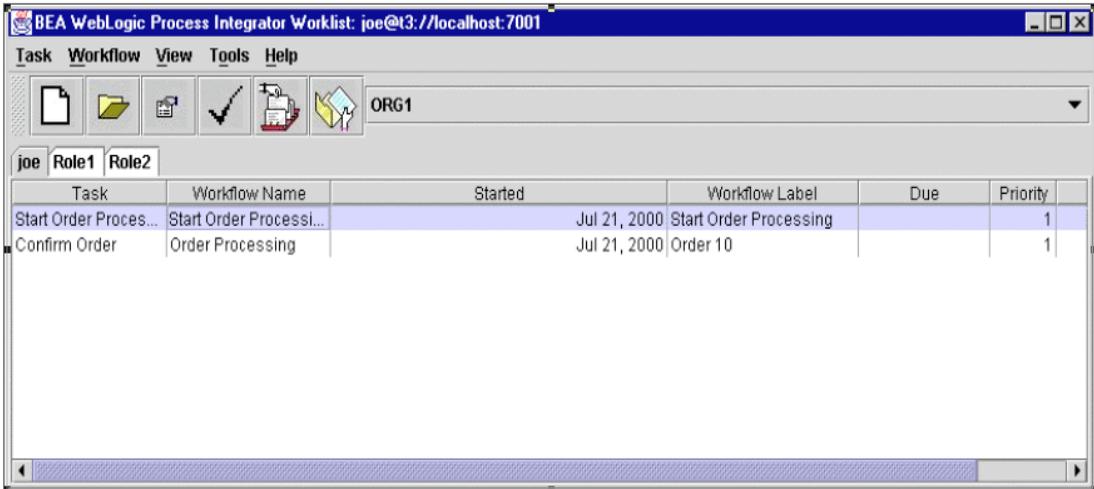
Figure 5-4 Worklist Application: Task List



4. Double-click the Start Order Processing task to execute the actions defined earlier under the Executed event of the Start Order Processing task; in this case, the creation of an XML document that triggers the Order Processing workflow.

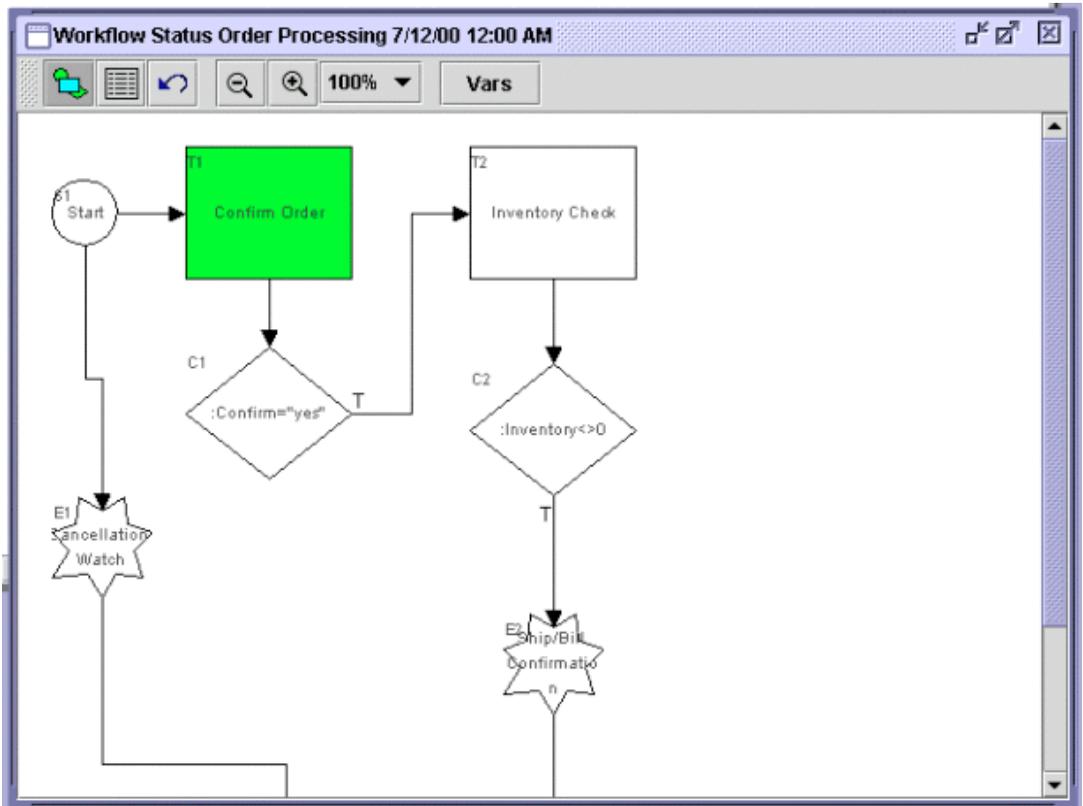
The first task of the Order Processing workflow is Confirm Order, which is assigned to the user: joe. This task is displayed on Joe’s worklist, once the workflow is triggered.

Figure 5-5 Worklist Application: Triggered Workflow



- To verify that the Order Processing workflow has been triggered, you open the instances window of that workflow. In the WebLogic Process Integrator Studio, right-click the workflow template definition for the Order Processing workflow and choose Instances from the pop-up menu. The following dialog box is displayed.

Figure 5-6 Workflow Status Dialog Box



The diagram shows the instance that has been started.

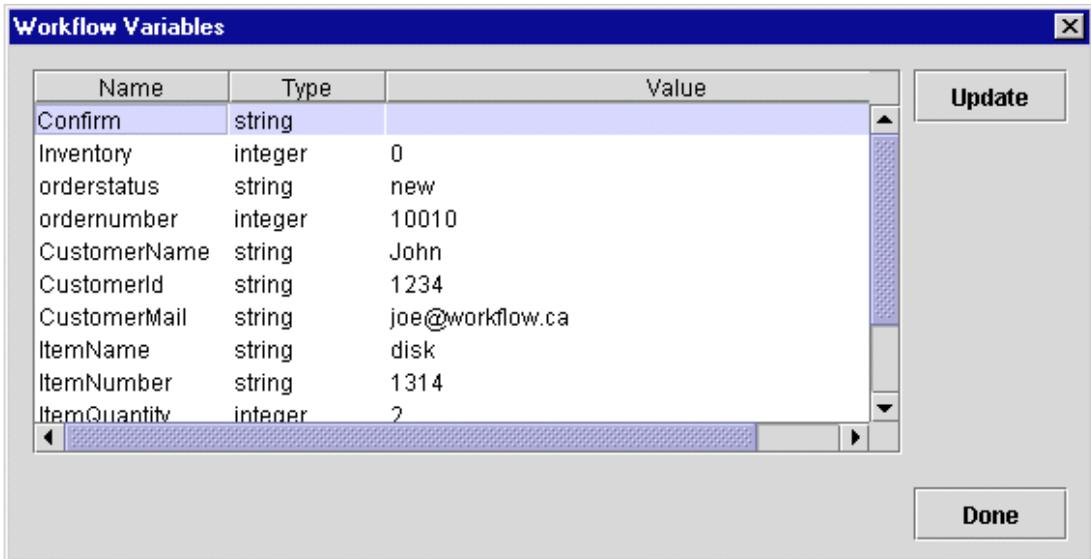
6. Right-click anywhere in the instance diagram to display a menu with the following options:
 - Workflow Status
 - Variables
 - Delete
7. Choose Workflow Status to find out which node is the current active one.

The Confirm Order task will be highlighted, showing that it is active and not yet executed.

8. Choose the Variables option to display the list of Order Processing workflow variables and their current values.

Note that all variables defined in the Start node are populated by the values coming from the XML document. The variable Confirm is still empty because the user has not confirmed the order yet.

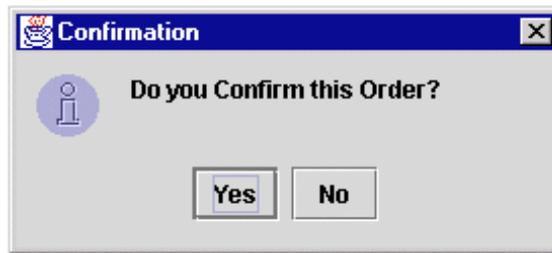
Figure 5-7 Workflow Variables Dialog Box



9. Return to the worklist and double-click the Confirm Order task.

This executes the Send XML to Client action defined under the Executed tab of the Confirm Order Task Properties dialog box. The action displays the Confirmation message box to the user joe.

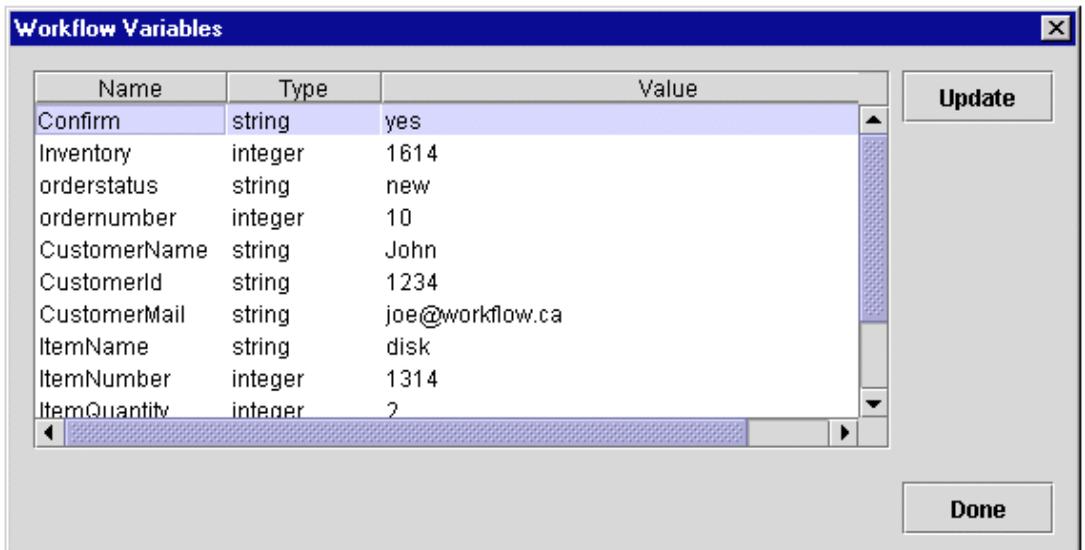
Figure 5-8 Confirmation Message



10. Click Yes to make the flow proceed and call the Inventory bean. Check the status of the Order Processing workflow again in the instances dialog box and particularly the values of the variables.

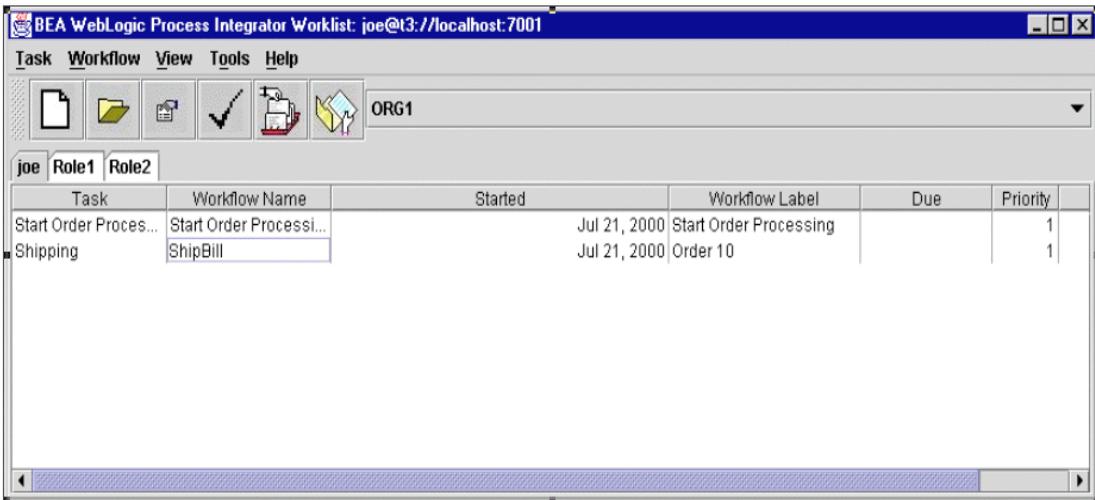
The dialog box shows that both Confirm and Inventory now have values.

Figure 5-9 Workflow Variables Dialog Box



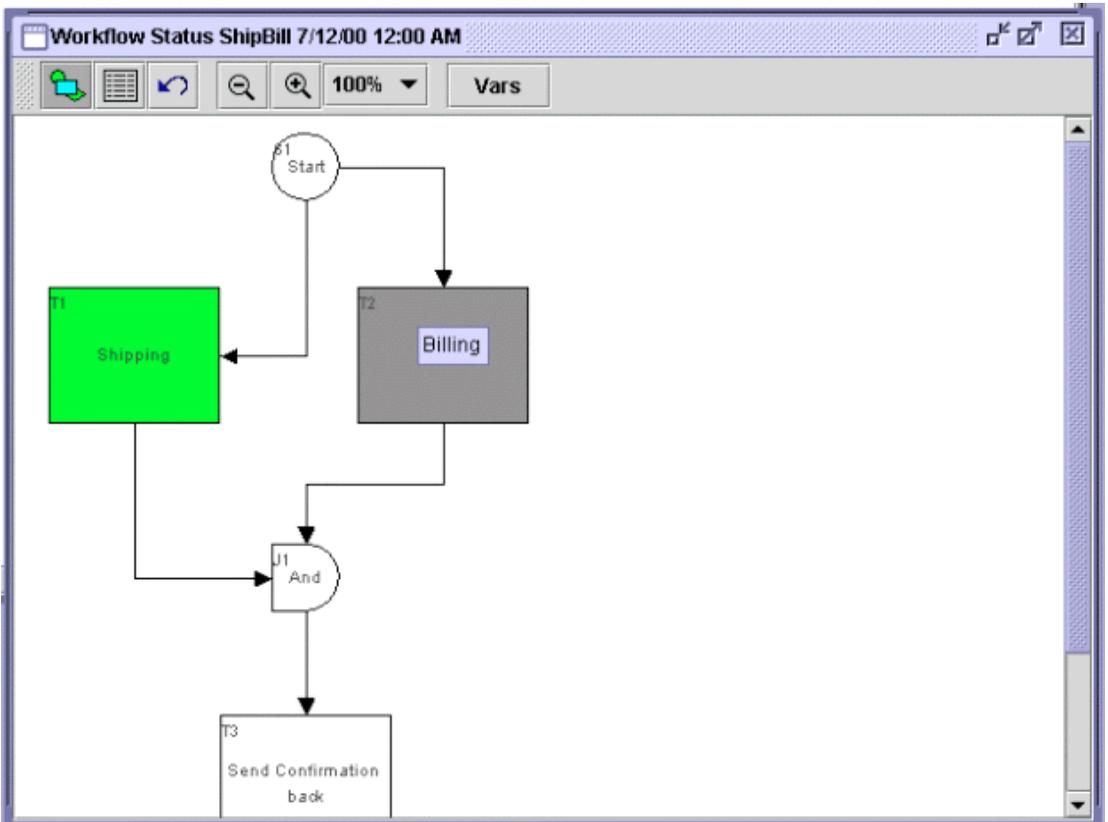
Because the value of the variable Inventory is not zero, the Ship/Bill workflow is called as the result of the execution of the Start Workflow action in the true case of the decision. This workflow assigns the Shipping task to the user joe. Therefore, joe's task list is updated with that task as follows:

Figure 5-10 Worklist Application: Joe's Task List



11. In the Studio application, right-click the ShipBill workflow template definition in the folder tree and choose Instances from the pop-up menu to display the started instance of the ShipBill workflow.

Figure 5-11 Workflow Status: ShipBill Workflow

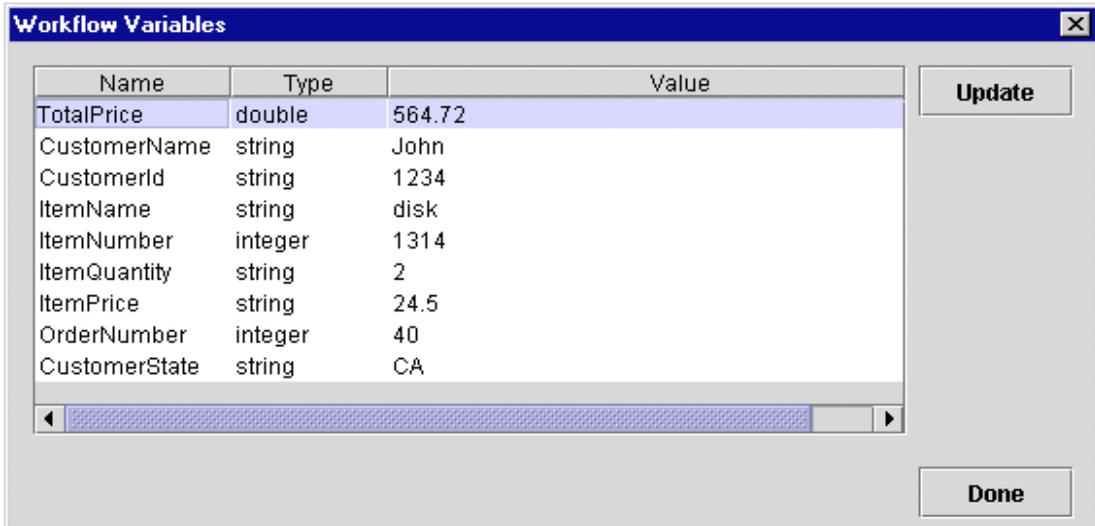


Note that the task Shipping is highlighted, because it is still active and has not been executed or marked done. However, the Billing task is marked as done because all actions related to it were defined within the Activated event. In other words, when the Billing task was executed, all actions (including Mark Task as Done) assigned in the Activated tab of the Billing task were executed.

Also note that the flow is waiting for the task Shipping to be executed in order to satisfy the AND join and move on.

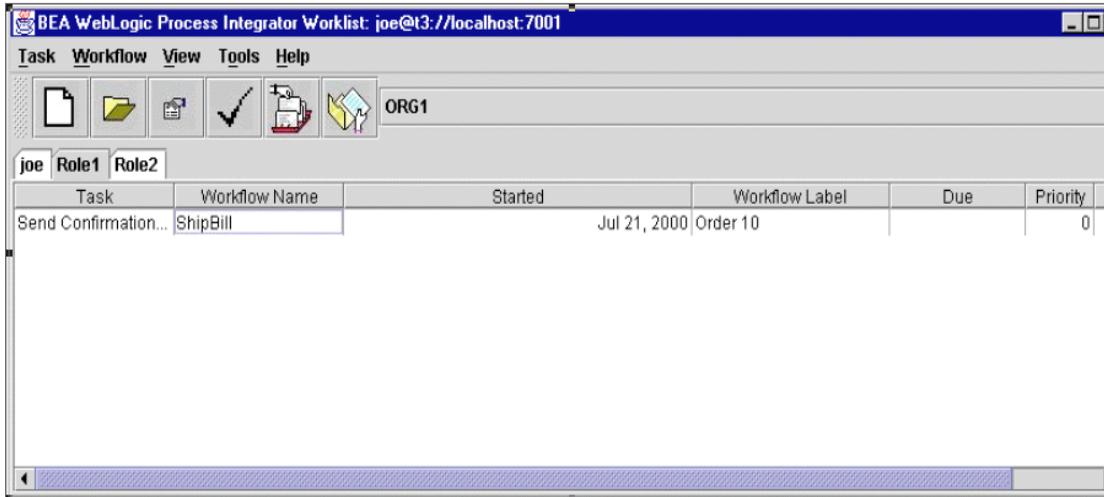
All variables that have been transferred from the parent Order Processing workflow are populated. The variable TotalPrice is calculated as the result of the call of the Billing POBean.

Figure 5-12 Workflow Variables Dialog Box



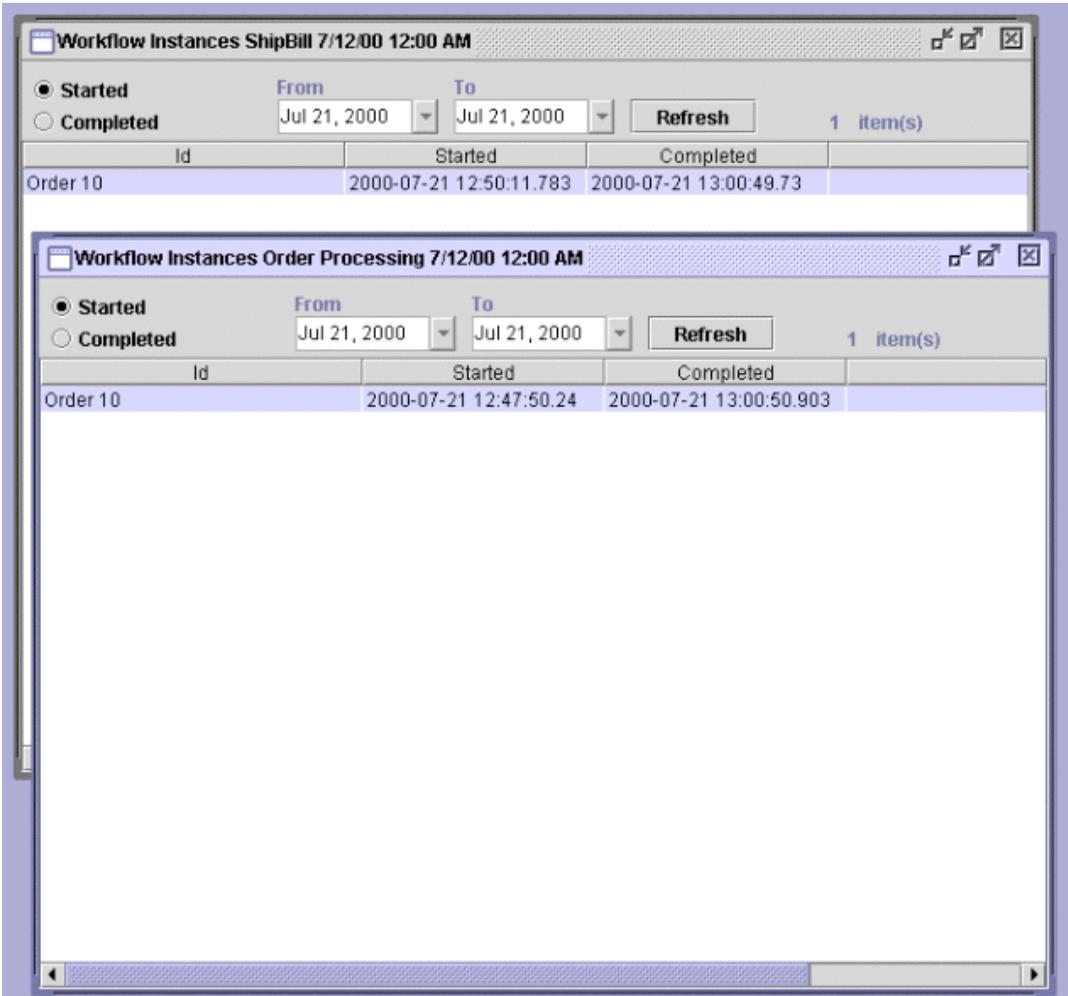
12. On the worklist, double-click the Shipping task to execute it. The ShipBill workflow proceeds, and the next task, Send Confirmation Back, is activated.
13. Send Confirmation Back is assigned to Role1. Select the Role1 tab on joe's worklist to see the task.

Figure 5-13 Worklist Application: Role1 Assignment



14. Double-click the Send Confirmation Back task to execute it and send an XML message back to the Order Processing workflow that completes the process.
The Order Processing and ShipBill workflow instances are marked as complete as reflected in the monitoring dialog boxes. The completed date and time are displayed.

Figure 5-14 Workflow Instances Dialog Boxes: Completed Date and Time



The preceding procedure is one scenario for this process. Another scenario that can be tested is to answer No to the confirmation and cause the order to cancel. In this case, the monitoring dialog box shows the instance of the Order Processing workflow with the comment Workflow Cancelled as defined in the Set Workflow Comment action.

Figure 5-15 Workflow Instances Dialog Box

Workflow Instances Order Processing 7/12/00 12:00 AM

Started
 Completed

From: Jul 20, 2000 To: Jul 21, 2000 Refresh 21 item(s)

Id	Started	Completed	
Order 10	2000-07-20 10:03:58.166	2000-07-20 10:47:14.686	Order Cancelled
Order 100	2000-07-20 10:55:39.383	2000-07-20 10:55:53.313	Order Cancelled
Order 110	2000-07-20 10:55:43.67	2000-07-20 10:55:54.036	Order Cancelled
Order 120	2000-07-20 10:55:46.133	2000-07-20 10:55:54.456	Order Cancelled
Order 130	2000-07-20 10:55:48.936	2000-07-20 10:55:54.796	Order Cancelled
Order 140	2000-07-20 10:56:15.866	2000-07-20 10:56:30.446	Order Cancelled
Order 150	2000-07-20 10:56:18.08	2000-07-20 10:56:30.716	Order Cancelled
Order 160	2000-07-20 10:56:23.796	2000-07-20 10:56:31.016	Order Cancelled
Order 170	2000-07-20 10:56:25.48	2000-07-20 10:56:31.31	Order Cancelled
Order 180	2000-07-20 11:02:07.353	2000-07-20 11:04:15.336	Order Cancelled
Order 190	2000-07-20 11:02:14.403	2000-07-20 11:04:15.856	Order Cancelled
Order 200	2000-07-20 11:04:26.693	2000-07-20 11:04:39.07	Order Cancelled
Order 210	2000-07-20 11:04:29.856	2000-07-20 11:04:39.35	Order Cancelled
Order 220	2000-07-20 11:04:56.866	2000-07-20 12:31:06.31	Order Cancelled
Order 230	2000-07-20 11:05:01.083	2000-07-20 12:31:06.61	Order Cancelled
Order 240	2000-07-20 11:05:13.23	2000-07-20 12:31:06.89	Order Cancelled
Order 250	2000-07-20 12:30:48.373	2000-07-20 12:31:07.24	Order Cancelled
Order 260	2000-07-20 12:30:54.0	2000-07-20 12:31:07.55	Order Cancelled
Order 270	2000-07-20 12:30:56.736	2000-07-20 12:31:07.87	Order Cancelled
Order 30	2000-07-20 17:46:35.276	2000-07-21 10:27:44.863	
Order 10	2000-07-21 12:47:50.24	2000-07-21 13:00:50.903	