IBM WebSphere Message Broker - Common Base Event Publisher

Version 1.1

Version 1.1, March 2008

This edition applies to Version 6.0.0.3, Version 6.1 of IBM WebSphere Message Broker, Version 6.0.2, Version 6.1 of IBM WebSphere Message Broker Toolkit, Version 6.0.2, Version 6.1 of IBM WebSphere Process Server.

Contents

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area.

Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION ″AS IS″ WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Department LZKS
11400 Burnet Road
Austin, TX 78758
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

The insurance quote software provided with this product is for use in validation of an IBM WebSphere Message Broker installation only, and is not to be used for any commercial purpose whatsoever, including but not limited to the provision of insurance quotations or any other insurance-related purpose.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

- IBM
- WebSphere

The following terms are trademarks of other companies:

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Acknowledgments

The Event Publisher Sub-flow is an IBM WebSphere Message Broker sub-flow that captures data from broker as Common Base Events. You can download a set of Common Base Event JAR files from the Test and Performance Tools Platform (TPTP) project of Eclipse for the Event Publisher Sub-flow, or use the set of Common Base Event JAR files that is re-distributed by this package. For more information about Common Base Event, see http://www.ibm.com/developerworks/library/specification/ws-cbe/. For more information about Test and Performance Tools Platform (TPTP) project of Eclipse, see http://www.eclipse.org/tptp/home/downloads/ .

Feedback and Support

All feedback for the package is welcome, but please be aware that a response is NOT guaranteed. Please contact SBPFBack@uk.ibm.com with your comments.

Please read the license file that accompanies the package to determine if you want to use it.

What is new?

The following enhancements have been made into Version 1.1 of this package:

- Fixed support to fetch data from the Global Environment section of a broker message.
- Changed creation time in generated CBE to be Coordinated Universal Time (UTC).
- Added millisecond precision to the creation time in the Common Based Event.
- Added capability to handle hexadecimal data in incoming message.
- Added capability to support multiple levels of sub-children when fetching a specified portion of the incoming message as an extended data element.
- Added the ability to support empty data in the incoming message.
- Added support to the Event-Publisher Message Driven Bean to handle CBE emitted as XML doc as well as element tree.

About this package

The IBM WebSphere Message Broker - Common Base Event Publisher supplies you with two reusable assets. The first reusable asset is an Event Publisher Sub-flow that captures data in IBM® WebSphere® Message Broker (WMB) as Common Base Event (CBE) and sends the events to a WebSphere MQ (WMQ) queue. The second asset is an Event Publisher Message Driven bean. By deploying this bean on IBM WebSphere Process Server (WPS), you can publish a Common Base Event from the WMQ queue to Common Event Infrastructure (CEI) of WPS.

Possible uses

Business activity monitoring enablement - Users can capture business data from broker message as Common Base Events using the Event Publisher Sub-flow. By deploying the Event Publisher Message Driven Bean, these business events can then be published from WebSphere Message Broker to Common Event Infrastructure of WebSphere Process Server. Users can simply navigate these business events using Common Base Event Browser of WebSphere Process Server or use IBM WebSphere Business Monitor for business activity monitoring.

In addition users have the flexibility to develop their own message driven bean and use the package for other business needs. For example:

- Security compliance enablement - Users can capture data from ROOT part of broker message as Common Base Events using the Event Publisher Sub-flow, and thus able to capture important information like security token identity from web service header for compliance management purposes.
- System activity monitoring enablement - Users can capture data from EXCEPTION part of broker message as Common Base Events using the Event Publisher Sub-flow, or associate the Event Publisher Sub-flow with failure terminal of broker nodes for system activity monitoring.

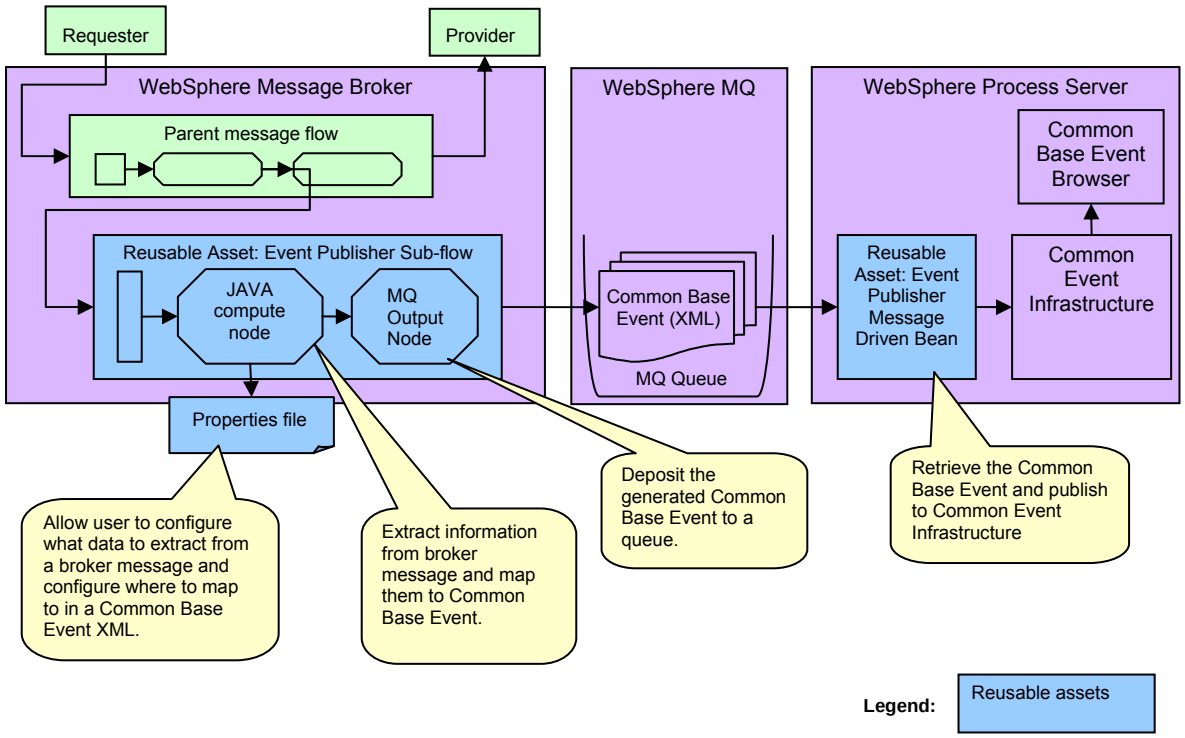
Prerequisites

The Event Publisher Sub-flow requires IBM WebSphere Message Broker version 6.0.0.3 or version 6.1, and IBM WebSphere Message Broker Toolkit version 6.0.2 or version 6.1. The Event Publisher Message Driven Bean requires IBM WebSphere Process Server version 6.0 or version 6.1. This package was developed for and tested in Microsoft® Windows®.

1 Introduction

This document describes an IBM WebSphere Message Broker Event Publisher that publishes data in IBM WebSphere Message Broker as Common Base Event to Common Event Infrastructure (CEI) of IBM WebSphere Process Server (WPS).

Figure 1. Publishing data in IBM WebSphere Message Broker as Common Base Event to the Common Event Infrastructure of IBM WebSphere Process Server



This package supplies you with two reusable assets:

- An Event Publisher Sub-flow that captures data in IBM WebSphere Message Broker as a Common Base Event. This sub-flow contains the following key nodes:
  - A Java™ compute node for extracting data from broker messages and mapping them into Common Base Event XML. You can configure the mapping information using the properties file.
  - An MQ output node that puts a Common Base Event XML message into a MQ queue.
- An Event Publisher Message Driven Bean that listens to the designated MQ queue for Common Base Event XML messages, retrieves them, and then publishes them to the Common Event Infrastructure of IBM WebSphere Process Server. The message driven bean deploys on IBM WebSphere Process Server.

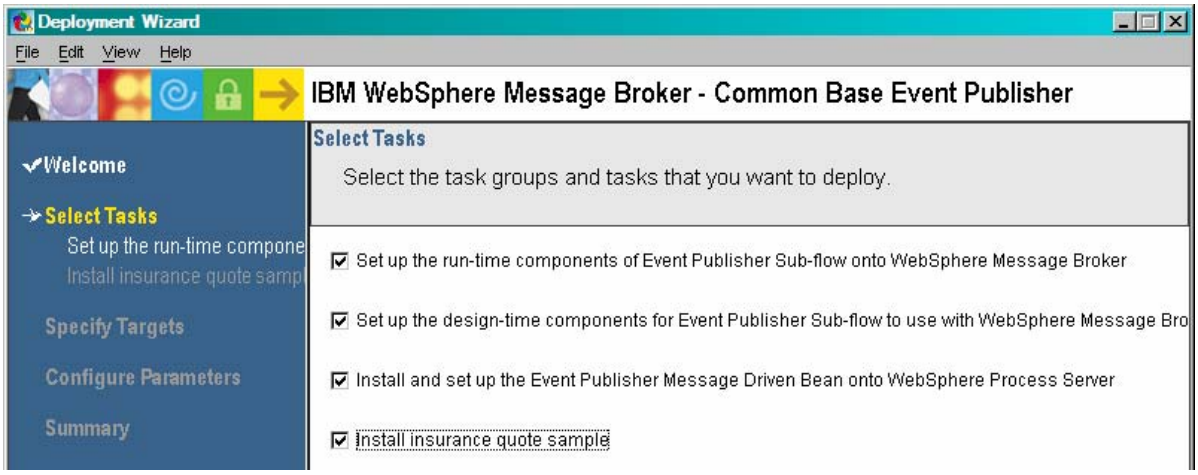
2 Installing the package

You can install the package either manually or using the installer wizard. If you use the wizard, certain default settings are selected automatically. For experienced users, see *Appendix C: Installing the package manually* to install the package manually using your own customized settings.

Before installing the package, you must have a broker configuration manager and a broker domain ready. For more information about creating broker domains, see the IBM WebSphere Message Broker Information Center.

To install the package:

1. Create a broker configuration manager and a broker domain.
2. Extract the package IA9V.zip to a temporary directory.
3. Run **Setup/WindowsSetup.exe**.
4. Select the number of tasks that you want to deploy. You can select all of the tasks listed and deploy the entire package all at once, or you can select and deploy each task separately. The following sections describe what each task does.



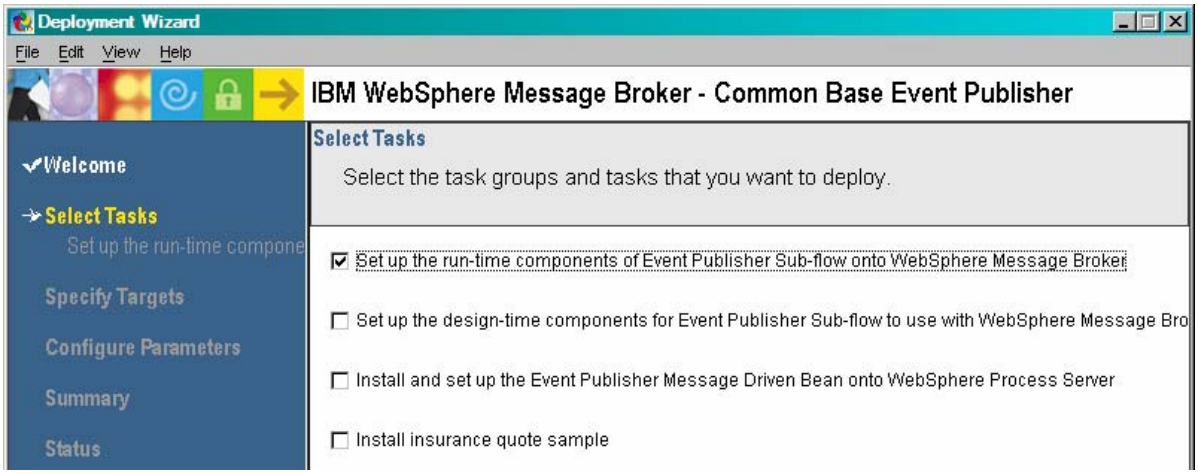
2.1 Event Publisher Sub-flow

The Event Publisher Sub-flow installation consists of two parts: setting up the run-time environment and setting up the design-time environment.

2.1.1 Setting up the run-time environment

The following steps walk you through the installation of the Event Publisher Sub-flow run-time environment. You need a queue manager that is associated with the created broker domain before installation.

1. Select **Set up the run-time components of Event Publisher Sub-flow onto WebSphere Message Broker**.



2. Select **Set up the run-time components of Event Publisher Sub-flow onto WebSphere Message Broker** to copy the Common Base Event JAR files and to create an MQ queue.
3. Select the manual task **Restart IBM WebSphere Message Broker (Manual Task)**.

4. Specify the target computer. **IBM WebSphere Message Broker** must be installed on the target computer. Click **Test Connection** to test if the target computer is available.
5. Specify values for the following fields:

Table 1. Run-time component typical fields and their default values

Typical fields	Default
*WebSphere Message Broker shared class folder for the Common Base Event JAR files	C:\Documents and Settings\All Users\Application Data\IBM\MQSI\shared-classes
Overwrite jar file if exists	No

Table 2. Run-time component advanced fields and their default values

Advanced fields	Default
*MQ queue name for storing generated Common Base Events	CBE_OUTPUT
*MQ queue manager name for storing and managing the above specified queue (must exist already)	WBRK6_DEFAULT_QUEUE_MANAGER
Create queue if do not exist	Yes

Note: * indicates a required field.

6. Click **Deploy all** to deploy all of the tasks.
7. Restart your IBM WebSphere Message Broker manually for this run-time to take effect. For more information about restarting a broker domain, see the IBM WebSphere Message Broker Information Center.
8. Select **Yes** and click **OK** to confirm the completion of the manual task.
9. Check the master log for detailed information about the deployment process.

2.1.2 Setting up the design-time environment

To set up the Event Publisher Sub-flow design-time environment, perform the following steps to put the development files (e.g. Event Publisher Sub-flow project interchange archived files) into your IBM WebSphere Message Broker Toolkit development environment.

1. Select **Set up the design-time components for Event Publisher Sub-flow to use with WebSphere Message Broker Toolkit**.
2. Specify the target computer. **IBM WebSphere Message Broker Toolkit** must be installed on the target computer. Click **Test Connection** to test if the target computer is available.
3. Specify values for the following required fields:

Table 3. Design-time component typical fields and their default values

Typical fields	Default
*Installation directory of the Event Publisher Sub-flow design-time components	C:\Program Files\IBM\IA9V

Table 4. Design-time component advanced fields and their default values

Advanced fields	Default
Create folder if do not exist	Yes
Overwrite file if already exist	Yes

Note: * indicates a required field.

4. Click **Deploy all** to deploy all of the tasks. Check the master log for detailed information about the deployment process.

2.2 Event Publisher Message Driven Bean

When you install the package, you also install the Event Publisher Message Driven Bean onto your IBM WebSphere Process Server. The following steps describe how to install and set up the Event Publisher Message Driven Bean using the installer.

2.2.1 Installing and setting up the Event Publisher Message Driven Bean

1. Select **Install and set up the Event Publisher Message Driven Bean onto WebSphere Process Server.**
2. Specify the target computer. **IBM WebSphere Process Server** must be installed on the target computer. Click **Test Connection** to test if the target computer is available.
3. Specify values for the following required fields:

Table 5. Event Publisher Message Driven Bean installation typical fields and default values

Typical fields	Default
*WebSphere Process Server installation directory	C:\Program Files\IBM\WebSphere\ProcServer
*WebSphere Process Server profile name	default
Restart server after installation	Yes

Table 6. Event Publisher Message Driven Bean installation advanced fields and default values

Advanced fields	Default
WebSphere Process Server administrator id	*[no default]*
WebSphere Process Server administrator password	*[no default]*
*MQ queue name for storing generated Common Base Events [2]	CBE_OUTPUT
*MQ queue manager name for storing and managing the above specified queue [2]	WBRK6_DEFAULT_QUEUE_MANAGER
*Listener port number for the queue manager above	2414
*Hostname where above queue manager resides [3]	localhost

Notes:

1. * indicates a required field.
2. MQ queue name and MQ queue manager name must match the names specified in *Section 2.1.1.* If both tasks are installed together, these 2 input fields are not editable and the values specified in *Section 2.1.1* are used instead.
3. “Hostname where queue manager resides” refers to where you installed IBM WebSphere Message Broker (the target computer in *Section 2.1.1)*.

4. Click **Deploy all** to deploy all of the tasks. Check the master log for detailed information about the deployment process.

2.3 Insurance quote sample

An insurance quote sample is included for you to validate your installation and for you to gain some experience using the package. There are three parts to the installation of the insurance quote sample:

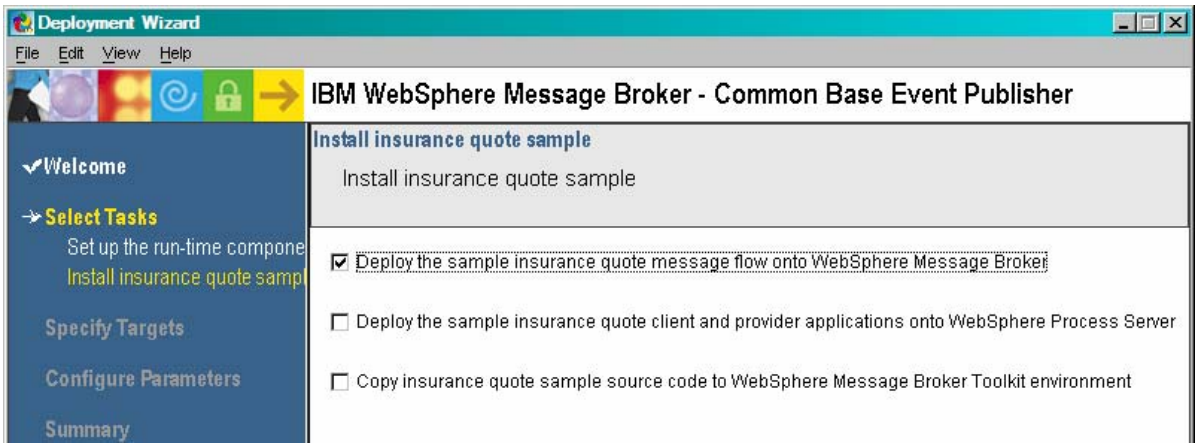
1. Deploy the sample insurance quote message flow onto IBM WebSphere Message Broker.
2. Deploy the sample insurance quote client and provider applications onto IBM WebSphere Process Server.
3. Copy insurance quote sample source code to IBM WebSphere Message Broker Toolkit environment.

See *Section 4 Testing with the insurance quote sample* for more information about this sample.

2.3.1 Deploying the sample insurance quote message flow

Before you deploy the sample insurance quote message flow, you must have a broker configuration manager and a broker domain ready for the sample message flow deployment. The following steps walk you through the deployment process for the insurance quote message flow in your IBM WebSphere Message Broker run-time environment.

1. Select **Install insurance quote sample**.
2. Select **Deploy the sample insurance quote message flow onto WebSphere Message Broker**.



(Optional) You can select and deploy all of the tasks and components at once. Read the following sections for details on the other sample installation tasks.

3. Specify the target computer. The **Run-time components for Event Publisher Sub-flow** must be installed on the target computer (see *Section 2.1.1*). Click **Test Connection** to test if the target computer is available.
4. Specify values for the following required fields:

Table 7. Typical input fields and their default values for the sample installation

Typical fields	Default
*WebSphere Message Broker installation directory	C:\Program Files\IBM\MQSI\6.0
*Broker name for the sample message flow (must exist already)	WBRK6_DEFAULT_BROKER
*Host name or IP address of the Configuration Manager	localhost
*Port number of the Configuration Manager	2414
*MQ queue manager that the Configuration Manager is using	WBRK6_DEFAULT_QUEUE_MANAGER

Table 8. Advanced input fields and their default values for the sample installation

Advanced fields	Default
*Execution group for the sample message flow	EventPublisherExeGrp
Create execution group if do not exist	Yes

Note: * indicates a required field.

5. Click **Deploy all** to deploy all of the tasks. Check the master log for detailed information about the deployment process.

2.3.2 Deploying the sample insurance quote client and provider applications

Perform the following steps to deploy the sample insurance quote client application and provider application to IBM WebSphere Process Server.

1. Select **Install insurance quote sample**.
2. Select **Deploy the sample insurance quote client and provider applications onto WebSphere Process Server**.
3. Specify the target computer. **IBM WebSphere Process Server** must be installed on the target computer. Click **Test Connection** to test if the target computer is available.
4. Specify values for the following required fields:

Table 9. Typical input fields and their default values for deploying the sample insurance quote client and provider applications

Typical fields	Default
*WebSphere Process Server installation directory [2]	C:\Program Files\IBM\WebSphere\ProcServer
*WebSphere Process Server profile name [2]	default

Table 10. Advanced input fields and their default values for deploying the sample insurance quote client and provider applications

Advanced fields	Default
WebSphere Process Server administrator id [2]	*[no default]*
WebSphere Process Server administrator password [2]	*[no default]*

Notes:

1. * indicates a required field.
2. The values for these four fields must match those specified in *Section 2.2.1*. If both tasks are installed together, these input fields might not be available and the values specified in *Section 2.2.1* are used instead.

5. Click **Deploy all** to deploy all of the tasks. Check the master log for detailed information about the deployment process.

2.3.3 Copying insurance quote sample source code

Perform the following steps to put the development files (e.g. insurance quote sample project interchange archived files) into your IBM WebSphere Message Broker Toolkit development environment.

1. Select **Install insurance quote sample**.
2. Select **Copy insurance quote sample source code to WebSphere Message Broker Toolkit environment**.
3. Specify the target computer. The **Design-time components for Event Publisher Sub-flow** must be installed on the target computer (see *Section 2.1.2*). Click **Test Connection** to test if the target computer is available.
4. Specify values for the following required fields:

Table 11. Typical input fields and default values for installing source code

Typical fields	Default
*Installation directory of the Event Publisher Sub-flow design-time components [2]	C:\Program Files\IBM\IA9V

Table 12. Advanced input fields and default values for installing source code

Advanced fields	Default
Create folder if do not exist [2]	Yes
Overwrite files if already exist [2]	Yes

Note:

1. * indicates a required field.
2. The values for these three fields must match the values specified in *Section 2.1.2*. If both tasks are installed together, these input fields might not be available and the values specified in *Section 2.1.2* are used instead.

5. Click **Deploy all** to deploy all of the tasks. Check the master log for detailed information about the deployment process.

3 Using the package

This package supplies you with two reusable assets for publishing Common Base Events from IBM WebSphere Message Broker to Common Event Infrastructure of IBM WebSphere Process Server.

3.1 Capture the events using the Event Publisher Sub-flow

3.1.1 Importing the Event Publisher Sub-flow into your workbench

To import the Event Publisher Sub-flow into your workbench:

1. Start IBM WebSphere Message Broker Toolkit.
2. Click **Window** > **Open Perspective** > **Broker Application Development**.
3. Click **File** > **Import**. In the import dialog, select **Project Interchange**, and then click **Next**.
4. Specify the location of **EventPublisherSubflow.zip** (e.g. *C:\Program Files\IBM\IA9V\EventPublisherSubflow.zip*) in the **From** zip file field and select all of the projects listed. This file was copied to the IBM WebSphere Message Broker Toolkit environment during the installation step in *Section 2.1.2*.

   Where:

   - CBESendFlow – is a message flow project that contains the Event Publisher Sub-flow.
   - CBESendFlowJava – is a Java project that contains all the code for the Java compute node used in the Event Publisher Sub-flow.
   - DefaultProp – contains the default properties file.

5. Click **Finish**. If you specified a different installation directory for the Event Publisher Sub-flow design-time components than the default value, then you might get a compilation error. The compilation error states that the Common Base Event JAR files are not found. Fix the build path to point to the directory where you installed the Event Publisher Sub-flow design-time components then rebuild the workspace.
6. *(Optional)* The Event Publisher Sub-flow deposits the generated Common Base Event onto a MQ queue (that holds these Common Base Events) called **CBE_OUTPUT**. Perform the following steps to modify the Event Publisher Sub-flow project if you chose a different queue name in *Section 2.1.1* or if you have another queue with the same name under another queue manager.

   a. Open the sub-flow by double-clicking **CBESenderFlow.msgflow** in the **CBESenderFlow** folder.
   b. Right-click on the MQOutput node named **CBE_OUTPUT** and click **properties**
   c. Change the name in the Queue Name field to match your queue name (this name is likely CBE_OUTPUT).
   d. Specify the queue manager in the Queue Manager Name field.
   e. Click **OK,** and then **Save** the message flow.

3.1.2 Adding the Event Publisher Sub-flow to a parent flow

For more information about adding a sub-flow to a parent flow, see the IBM WebSphere Message Broker Information Center. You can capture the same set of data at different points of a parent flow by adding the Event Publisher Sub-flow multiple times.

To capture different sets of data at different points of a parent flow, you must add the Event Publisher Sub-flow multiple times and use different flow-level properties files. See *Appendix B* for more information.

3.1.3 Determining the set of data to be captured

Developers must work with business analysts to determine what data to capture. The Event Publisher Sub-flow supports two levels of properties file: the Default level and the Flow level.

Capturing a default set of data

Use the Default properties file to capture the general flow data into IBM WebSphere Message Broker. The default properties file specifies the default set of data to be captured for all instances of the Event Publisher Sub-flow.

Package the Default properties file as a JAR file and deploy it to the IBM WebSphere Message Broker shared class folder (e.g. *C:\Documents and Settings\All Users\Application Data\IBM\MQSI\shared-classes*). A version of the Default properties file is already deployed during the setup step in *Section 2.1.1*.

To modify and redeploy the Default properties file:

1. Modify the **props/Default.properties** file in the **DefaultProp** project of the IBM WebSphere Message Broker toolkit. Add/modify mapping entries in the file. For more information on mapping entries, see **How to map data to Common Base Event XML** in *Appendix A*.
2. Right-click the project and click **export** after you define the mapping information.
3. Export the project as a **jar file**.
4. Pick any name for the JAR file. The name used in this example is **defaultProp.jar**. Export this JAR file to any location.
5. Delete the **defaultProp.jar** file in the IBM WebSphere Message Broker shared class folder (e.g. *C:\Documents and Settings\All Users\Application Data\IBM\MQSI\shared-classes*). You might need to stop message broker and shut down all MQ related services before deleting the file.
6. Copy the newly created **defaultProp.jar** file to the IBM WebSphere Message Broker shared class folder.
7. Restart message broker.

Capturing flow-level specific data

You can configure every parent flow that includes the Event Publisher Sub-flow to capture a specific set of data using the flow-level properties file. If the flow-level properties file exists for a parent flow, then the default properties file is not used.

The flow-level properties file observes the following naming convention: "props/*SAL[parentflowname].properties*". For example, parent flow "InsuranceQuoteFlowRouting" can have a flow-level properties file "props/SALInsuranceQuoteFlowRouting.properties". Package the flow-level properties file as a JAR file and deploy it along with the parent flow.

To create a flow-level properties file for capturing flow-level specific data:

1. In the Java perspective, click **File** > **New** > **Project**.
2. Expand the **Java** folder and click **Java Project** then click **Next**.

3. Enter a name in the Project name field. For example: **InsuranceQuoteFlowProp** then click **Finish.**
4. Select your newly created project. Right-click and click **New** > **Folder**. Enter **props** in the Folder name field then select the newly created folder. Right-click and click **New** > **File**.
5. Enter a file name with **SAL** as the prefix, followed by the name of the parent flow. The extension is **properties**. For example, name a flow-level properties file for a parent flow **InsuranceQuoteFlowRouting, SALInsuranceQuoteFlowRouting.properties,** and then click **Finish**.
6. Modify the newly created properties file with mapping entries. For more information on mapping entries, see **How to map data to Common Base Event XML** in *Appendix A*.
7. Right-click the project and click **export**. Export the project as a **jar file.**
8. Pick any name for the JAR file. Export this JAR file to any location.
9. Launch Windows Explorer and locate the exported JAR file. Drag-and-drop the JAR file from that location into the corresponding parent flow project.

3.1.4 Deploying the parent flow

For more information about deploying a message flow, see the IBM WebSphere Message Broker Information Center. If you use the flow-level properties file, make sure that you include the JAR file of the flow-level properties file in the BAR file of the parent flow.

3.2 Publish the events - Using the Event Publisher Message Driven Bean

Make sure that you start the Event Publisher Message Driven Bean (installed in *Section 2.2.1)*.

1. Login to the WAS Admin Console.
2. Click **Applications – Enterprise Applications**. Make sure the **CEILoggingEAR** is started.

The following table describes the Common Event Infrastructure settings for the Event Publisher Message Driven Bean:

Table 13. Common Event Infrastructure settings

Common Event Infrastructure setting	Value	Variable in deployment descriptor
JNDI Name of the CEI Emitter Factory	com/ibm/events/configuration/emitter/Default	CEI_EMITTER_JNDI_NAME
CEI Transaction mode	DEFAULT	CEI_TRANSMODE
CEI Synchronous mode	DEFAULT	CEI_SYNCHMODE

If you want to change the CEI settings, you must import the **CEILoggingEAR.ear** file to the workbench and modify the values of the variables defined in the deployment descriptor (*ejb-jar.xml*). After you change the CEI settings, rebuild and redeploy the EAR file manually.

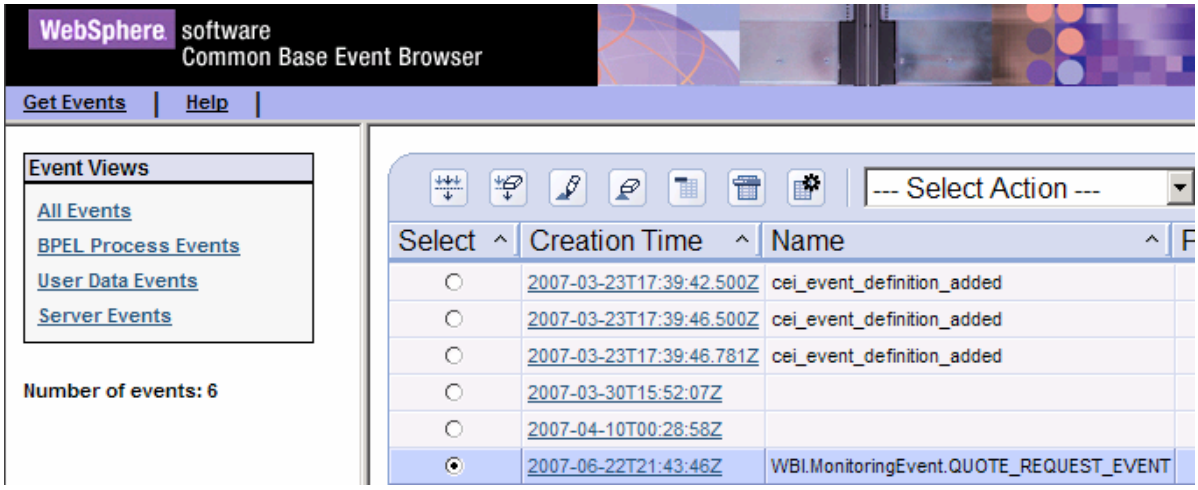
3.3 View the events

3.3.1 Using the Common Base Event Browser of IBM WebSphere Process Server

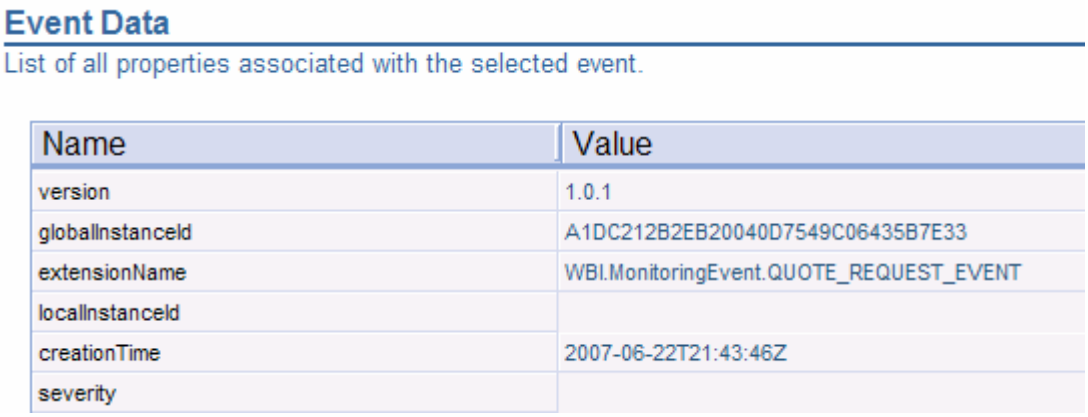
The Common Base Event Browser is a Web application that is included with IBM WebSphere Process Server. You can use the Common Base Event Browser to select, sort, and view events that are stored in the Common Event Infrastructure.

To view the events using the Common Base Event Browser:

1. Open a Web browser and enter the following address: http://*<apphost>:<appport>*/ibm/console/cbebrowser.
2. Click the “Get Events” button.
3. Click the "All Events" view from the navigator.



4. Select an event to view the details.



3.3.2 Using IBM WebSphere Business Monitor

IBM WebSphere Business Monitor is a tool to monitor business performance. It collects business performance-relevant data by extracting and aggregating information in business events. The

operation of IBM WebSphere Business Monitor is controlled by monitor models. You can construct a monitor model to monitor the business events that are generated by this package for monitoring business activity in IBM WebSphere Message Broker. Please refer to IBM WebSphere Business Monitor Information Centre for more information about constructing monitor model.

Please note that the events generated by the Event Publisher Sub-flow must contain the extension name element for correlation by the monitor model of IBM WebSphere Business Monitor.

Here is an example of a mapping entry:

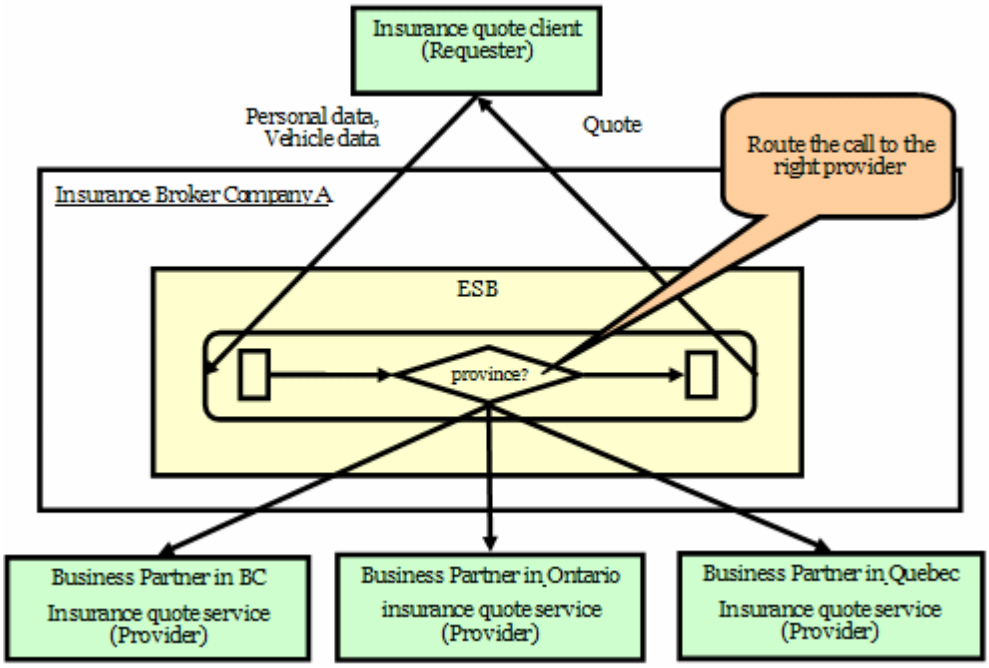
```
DATA_MAPTO_8    = EXTENSION_NAME
DATA_REL_PATH_8 = WBI.MonitoringEvent.QUOTE_REQUEST_EVENT
DATA_LOCATION_8 = CONSTANT
```

4 Testing with the insurance quote sample

4.1 Introduction

You can use the included insurance quote sample to validate your installation and get some experience using the package. This sample illustrates how you can capture the insurance quote information as a Common Base Event, publish the event to the Common Event Infrastructure and view it in the Common Base Event Browser or monitor it using IBM WebSphere Business Monitor.

Figure 2. Working with the insurance quote example



The insurance quote sample consists of 3 components:

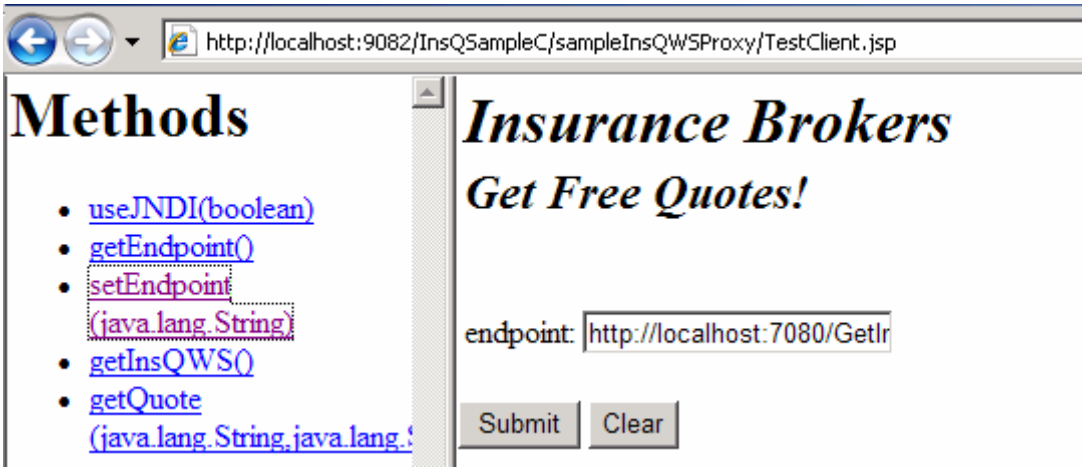
1. Insurance quote client application – a Web service requester (deployed on IBM WebSphere Process Server) that sends user requests on insurance quotes. User personal data (e.g. province information) along with vehicle data are included in the Web service request.
2. Insurance quote message flow – a message flow (deployed on IBM WebSphere Message Broker) that routes Web service requests to the appropriate insurance quote provider based on province information. The reusable asset Event Publisher Sub-flow has been inserted in this message flow to capture information about the province and the returned quote price as an event.
3. Insurance quote provider applications – 3 Web service providers (deployed on IBM WebSphere Process Server) that provide insurance quotes according to preset provincial regulations. 3 provinces are supported: British Columbia, Ontario and Quebec.

You can install the sample insurance quote client application and provider application to IBM WebSphere Process Server by following the installation in *Section 2.3.2*. The sample insurance quote message flow deploys to the IBM WebSphere Message Broker run-time by following the installation in *Section 2.3.1*.

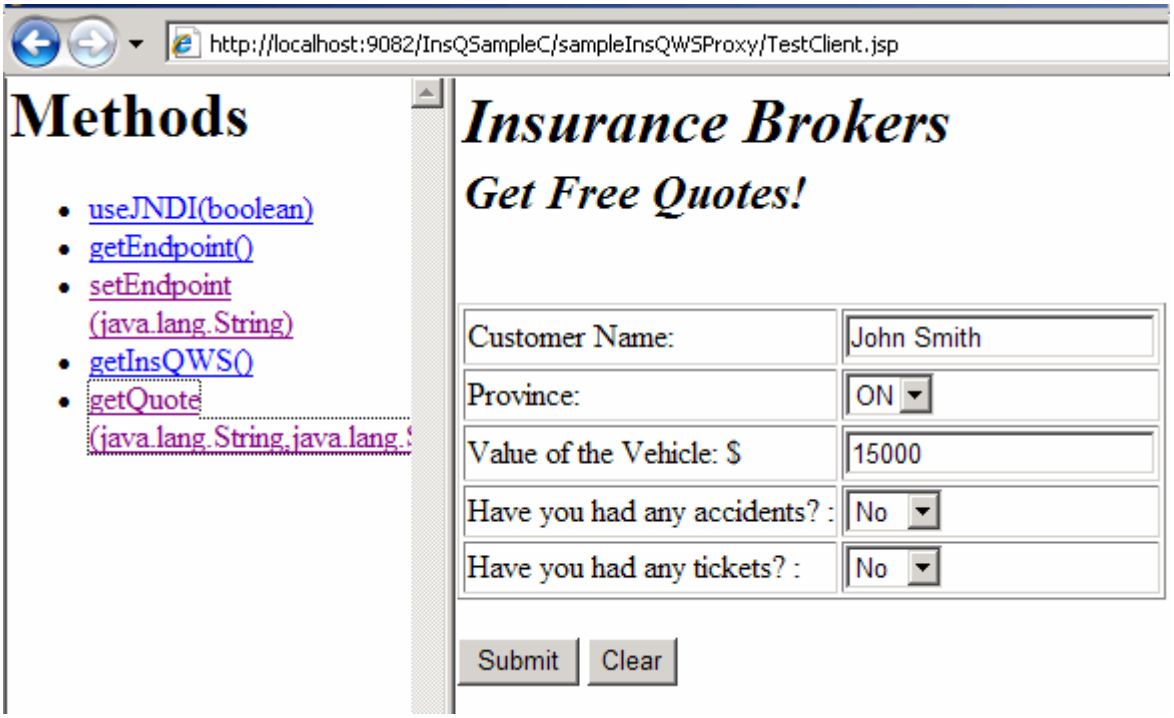
To understand the underneath implementation of this sample insurance quote message flow, follow *Section 2.3.3* to copy the source code of this sample to your IBM WebSphere Message Broker Toolkit development environment.

4.2 Generating the event

1. Open a Web browser and enter: http://<apphost>:<appport>/InsQSampleC/sampleInsQWSProxy/ TestClient.jsp.
2. Click **setEndpoint**. Specify the endpoint URL for the insurance quote message flow (http://<mbhost>:<mbport>/GetInsuranceWMB). For example, http://localhost:7080/GetInsuranceWMB.



3. Click **getQuote**. Enter the user information and click **Submit.**

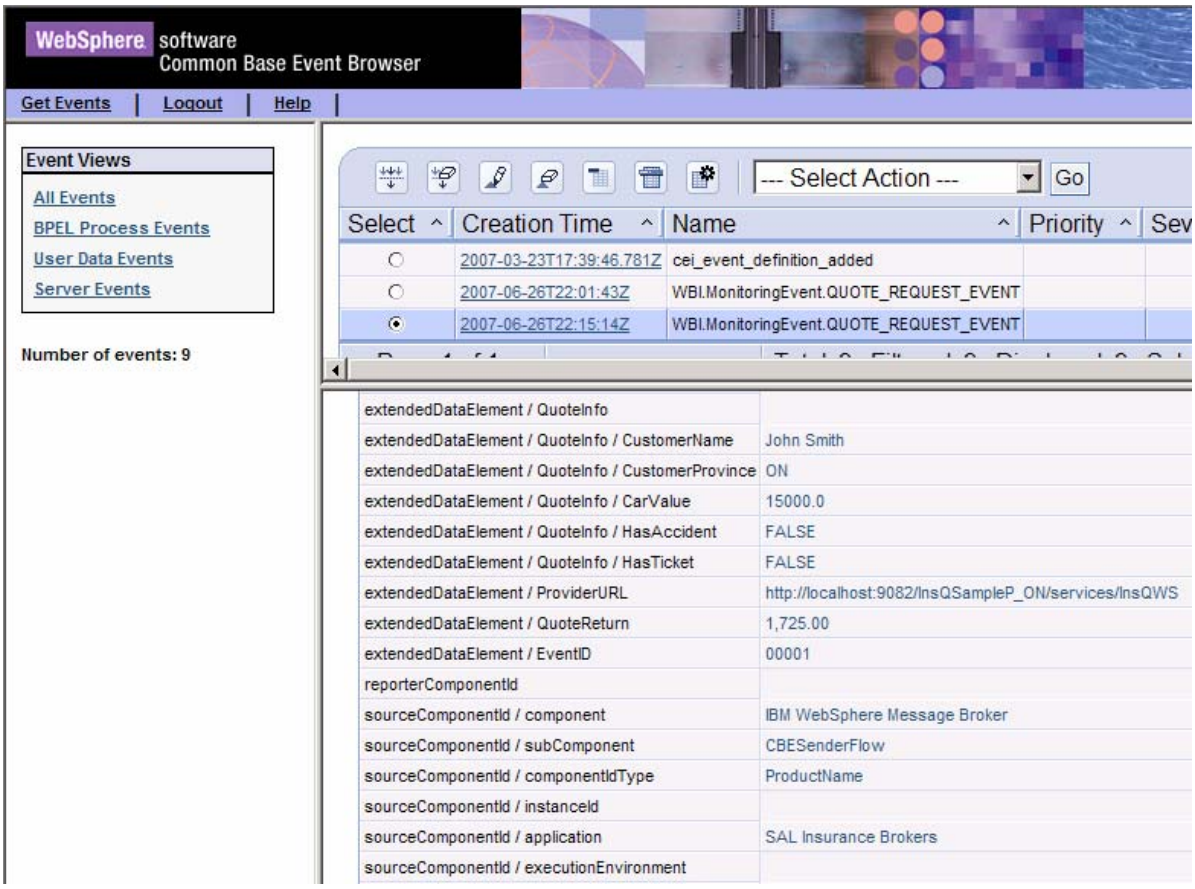


4. The insurance quote sample assumes that the IBM WebSphere Message Broker and IBM WebSphere Process Server reside on the same server, and the insurance quote provider application belongs to port number 9082. If you get a 404 error saying “endpoint not found”, your environment might contain different information than these assumptions. Follow *Section 4.4* to modify the message flow code and redeploy the message flow.

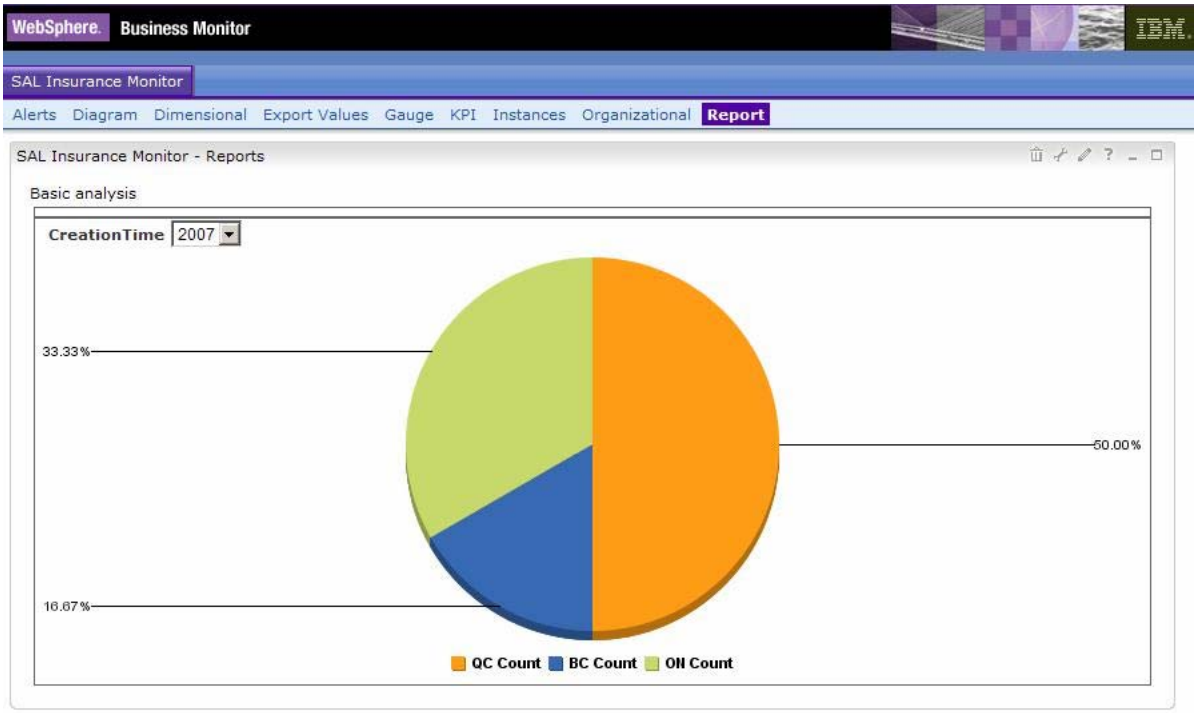
4.3 Viewing the event for this quote

The Event Publisher Sub-flow generates an event every time the user requests a quote. You can use the Common Base Event Browser to see these events.

1. Open a Web browser and enter the following address: http://*<apphost>:<appport>*/ibm/console/cbebrowser.
2. Click the “Get Events” button.
3. Click the "All Events" view from the navigator.
4. Select the latest generated Common Base Event. It might look similar to the following screen capture:



Or you can use IBM WebSphere Business Monitor to collect business performance-relevant data by extracting and aggregating information in these events. For example, the following diagram shows a pie chart that counts the number of insurance quotes handled by each provider (ON, BC, and QC). Please refer to IBM WebSphere Business Monitor Information Centre for more information about constructing monitor model.



4.4 Understanding the implementation

Assume that you have imported the Event Publisher Sub-flow project interchange file already. To look at the source code of this sample insurance quote application:

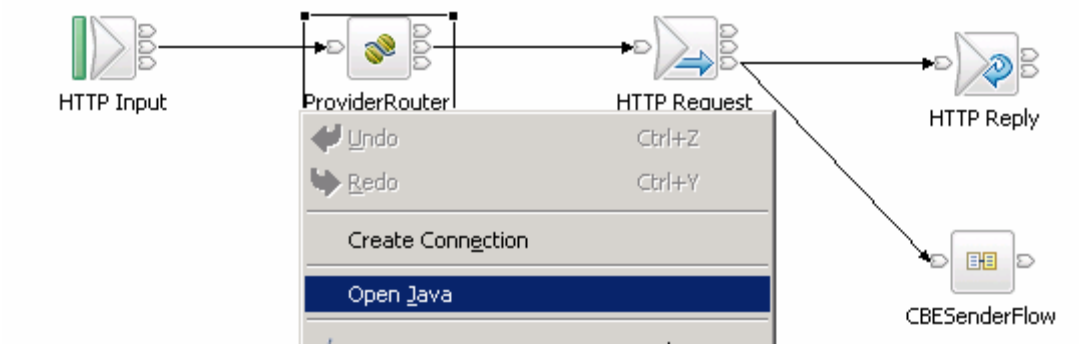
1. Start the IBM WebSphere Message Broker Toolkit
2. Click **Window** > **Open Perspective** > **Broker Application Development**.
3. Click **File** > **Import**. In the import dialog, select **Project Interchange**, and then click **Next**.
4. Specify the location of **InsQMsgFlow.zip** (e.g. *C:\Program Files\IBM\IA9V\InsQMsgFlow.zip*) in the **From** zip file field and select all projects listed. This file was copied to the IBM WebSphere Message Broker Toolkit environment during the installation step specified in *Section 2.3.3*.

   Where:

   - InsuranceQuoteFlow - a message flow project that contains the sample insurance message flow.
   - InsuranceQuoteFlowJava – a Java project that contains all the code for the Java compute node. This code is used in the sample insurance message flow.
   - InsuranceQuoteFlowProp – a flow-level properties file with mapping information that captures insurance quote information and converts it to Common Base Events.
   - InsQSampleC – a Web service client project that contains the insurance quote Web service client.
   - InsQSampleP_BC - a Web service project that contains the insurance quote Web service provider for British Columbia.
   - InsQSampleP_ON - a Web service project that contains the insurance quote Web service provider for Ontario.
   - InsQSampleP_QC - a Web service project that contains the insurance quote Web service provider for Quebec.

5. Click **Finish** and wait until the workspace is built.
6. Look at the insurance quote message flow to see how the Event Publisher Sub-flow is inserted.

    a. Modify the routing Java code if you get a 404 error in *Section 4.2* saying “endpoint not found”.
    b. Right-click the **ProviderRouter** Java Compute Node and click **Open Java**



    c. Modify the host name and port number. **HOSTNAME** is the name of the host of the IBM WebSphere Process Server (where you installed the insurance quote provider). **PORT** is the port number of the insurance quote provider applications. The following code sample represents these modifications:

```
protected static final String HOSTNAME = "localhost";
protected static final String PORT     = "9082";
```

    d. Save the changes the rebuild and redeploy the BAR file of the message flow.

7. Look at ProviderRouter to understand how we store different data from the incoming message to local environment and / or global environment. Then look at the insurance quote flow-level properties file to understand how mapping information captures insurance quote information as Common Base Events.
8. A Common Base Event similar to the following example will be generated:

```
<CommonBaseEvent creationTime="2008-02-14T19:40:52.453Z"
extensionName="WBI.MonitoringEvent.QUOTE_REQUEST_EVENT"
globalInstanceId="A1DCDB34C040D950FEB1874FF72C59CD" version="1.0.1">
        <extendedDataElements name="QuoteInfo" type="noValue">
                <children name="CustomerName" type="string">
                        <values>John</values>
                </children>
                <children name="CustomerProvince" type="string">
                        <values>ON</values>
                </children>
                <children name="CarValue" type="string">
                        <values>15000.0</values>
                </children>
                <children name="HasAccident" type="string">
                        <values>FALSE</values>
                </children>
                <children name="HasTicket" type="string">
                        <values>FALSE</values>
                </children>
        </extendedDataElements>
```

```
        <extendedDataElements name="ProviderURL" type="string">
                <values>http://localhost:9082/InsQSampleP_ON/services/InsQWS</values>
        </extendedDataElements>
        <extendedDataElements name="QuoteReturn" type="string">
                <values>1,725.00&#x9;</values>
        </extendedDataElements>
        <extendedDataElements name="EventID" type="string">
                <values>00001</values>
        </extendedDataElements>
        <extendedDataElements name="QuoteID" type="hexBinary">
                <hexValue>494e53515731323334</hexValue>
        </extendedDataElements>
        <extendedDataElements name="AdditionalCustomerData" type="noValue">
                <children name="CustomerProfile" type="noValue">
                        <children name="PersonalInformation" type="noValue">
                                <children name="Name" type="string">
                                        <values>John</values>
                                </children>
                                <children name="Age" type="string">
                                        <values>25</values>
                                </children>
                        </children>
                        <children name="DriverHistory" type="noValue">
                                <children name="NumberOfAccidents" type="string">
                                        <values>3</values>
                                </children>
                                <children name="LicenseStatus" type="noValue">
                                        <children name="Level" type="string">
                                                <values>G</values>
                                        </children>
                                        <children name="Expiry" type="string">
                                                <values>Dec2020</values>
                                        </children>
                                </children>
                        </children>
                </children>
        </extendedDataElements>
        <extendedDataElements name="ProviderURL - Ontario" type="string">
                <values>InsQSampleP_ON/services/InsQWS</values>
        </extendedDataElements>
        <extendedDataElements name="ProviderURL - British Columbia" type="string">
                <values>InsQSampleP_BC/services/InsQWS</values>
        </extendedDataElements>
        <extendedDataElements name="ProviderURL - Quebec" type="string">
                <values>InsQSampleP_QC/services/InsQWS</values>
        </extendedDataElements>
        <sourceComponentId application="SAL Insurance Brokers" component="IBM WebSphere
Message Broker" componentIdType="ProductName" location="localhost:7080"
locationType="Hostname" subComponent="CBESenderFlow"
componentType="http://www.ibm.com/namespaces/autonomic/ESB_componentTypes"/>
        <situation categoryName="ReportSituation">
                <situationType xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="ReportSituation" reasoningScope="INTERNAL" reportCategory="LOG"/>
        </situation>
</CommonBaseEvent>
```

Note: The extension name “extensionName="WBI.MonitoringEvent.QUOTE_REQUEST_EVENT" is needed for the monitor model of IBM WebSphere Business Monitor to correlate the events and generate business performance-relevant data.

5 Troubleshooting

5.1 Event Publisher Sub-flow

Currently, any exceptions that occur in the sub-flow are considered “soft” errors and do not impact the normal business process of the parent flow.

Expected application exceptions are handled by the sub-flow which logs them as warning messages in the system event log. Unexpected application exceptions are handled by the UnexpectedError trace node which logs them as exceptions in the system event log.

5.1.1 *Hints and tips*

Unable to deploy a message flow

Deploying a message flow takes time, but does not take over 3 minutes. If it takes longer, check the system event log for logged errors. There are several reasons why deployment can fail. If the log indicates a *ClassDefNotFound* error, make sure that the Common Base Event JAR files reside in the IBM WebSphere Message Broker shared class folder and the IBM WebSphere Message Broker has been restarted.

Changing flow-level properties file

Sometimes, when you deploy the new flow-level properties file, the system cannot read the file. To correct this problem, restart the Message Broker and Configuration Manager, and redeploy the same BAR file.

Reference

More information about IBM WebSphere Message Broker can be found in the following links:

- IBM WebSphere Message Broker Information Center, http://publib.boulder.ibm.com/infocenter/wmbhelp/v6r0m0/index.jsp
- IBM WebSphere Message Broker Basics Redbooks, http://www.redbooks.ibm.com/abstracts/SG247137.html?Open

5.2 Event Publisher Message Driven Bean

The SystemOut log of IBM WebSphere Process Server records all error messages from the Event Publisher Message Driven Bean.

5.2.1 Hints and tips

Permission error when security is turned on

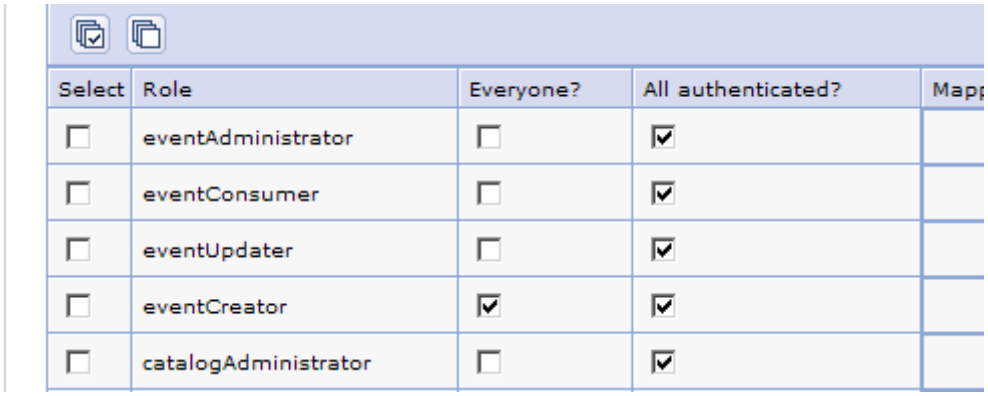
The following global security error requires you to take steps to grant permission to create events in the Common Event Infrastructure of IBM WebSphere Process Server:

```
SECJ0053E: Authorization failed for /UNAUTHENTICATED while invoking
(Bean)ejb/com/ibm/events/bus/EventBus
```

```
createEvent(org.eclipse.hyades.logging.events.cbe.CommonBaseEvent):3
securityName: /UNAUTHENTICATED;accessID: UNAUTHENTICATED is not
granted any of the required roles: eventAdministrator eventCreator
catalogAdministrator
```

To grant permission to create an event in the Common Event Infrastructure of IBM WebSphere Process Server:

1. Log on to admin console.
2. Select **Enterprise Applications**.
3. Select the application **EventServer**.
4. Select **Additional Properties** > **Map security roles to users/groups**.
5. Grant **EveryOne** access to the role **eventCreator** then click **OK** and save the changes.



Select	Role	Everyone?	All authenticated?	Mapp
☐	eventAdministrator	☐	☑	
☐	eventConsumer	☐	☑	
☐	eventUpdater	☐	☑	
☐	eventCreator	☑	☑	
☐	catalogAdministrator	☐	☑	

DuplicateHomeNameException

Occasionally, after installing the message driven bean multiple times, you might get the following exception in the *SystemOut.log*:

com.ibm.ejs.container.DuplicateHomeNameException

The following procedure will rectify this error:

1. Uninstall the Event Publisher Message Driven Bean and stop the server.
2. Enter the following directory: *<WPS_ProfilePathfortheEventPublisherMessageDrivenBean>\wstemp\events,* and delete the three *eventbufferX.ser* files.
3. Restart the server then reinstall the Event Publisher Message Driven Bean.

Reference

- IBM WebSphere Business Process Management information center, http://publib.boulder.ibm.com/infocenter/dmndhelp/v6rxmx/index.jsp
- Best Practices for the Common Base Event and Common Event Infrastructure, ftp://www6.software.ibm.com/software/developer/library/autonomic/books/cbepractice/index.htm

6 Appendix A

6.1 Mapping data to Common Base Event XML

The reusable asset Event Publisher Sub-flow captures data as Common Base Events and sends the events to a MQ queue. It extracts data from broker messages and maps them to Common Base Event XML using mapping entries in a properties file. Each mapping entry contains three properties:

DATA_MAPTO_* This property indicates which Common Base Event field you can map to. For more information, see the **ReadMe.txt** file in the DefaultProp project.

DATA_REL_PATH_* This property contains a constant value if `DATA_LOCATION_*` is specified as `CONSTANT`. Otherwise, it contains an XPath to locate the source data from the specified DATA_LOCATION_* within a broker message.

DATA_LOCATION_* This property indicates the location of the source data. A broker message contains different parts: a root, a message body, a local environment, a global environment and an exception list. This property specifies where in the broker message to extract the data from. There are six data locations that you can specify:

Table 14. Data locations

ROOT	Source data comes from the root of the broker message
MESSAGE_BODY	Source data comes from the message body part of the broker message
LOCAL_ENVI	Source data comes from the local environment of the broker message
GLOBAL_ENVI	Source data comes from the global environment of the broker message
EXCEPTION	Source data comes from the exception part of the broker message
CONSTANT	Source data is a constant value

Where * is an integer starting at 1. This index must be incremented continually. Although the order is not important, if you miss an index number, the system will consider it the end of the properties file.

The format of the properties file is not validated by the sub-flow because of the performance cost. Users may write their own tool to catch formatting problems such as missing index numbers or duplicate index numbers.

For example, the following code represents what a broker message might look like:

```
<soapenv:Envelope xmlns:soapenv=http://schemas.xmlsoap.org/soap/envelope/
        xmlns:soapenc=http://schemas.xmlsoap.org/soap/encoding/
        xmlns:xsd=http://www.w3.org/2001/XMLSchema
        xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance>
        <soapenv:Header/>
        <soapenv:Body>
            <p490:getQuoteResponse >
                <getQuoteReturn>1,725.00</getQuoteReturn>
            </p490:getQuoteResponse>
         </soapenv:Body>
</soapenv:Envelope>
```

The following mapping entries illustrate how you can capture the data from “getQuoteReturn” to extendedDataElements of a Common Base Event

```
DATA_MAPTO_8    = EDE_DATA.0.VALUES.0
DATA_REL_PATH_8 = Envelope/Body/getQuoteResponse/getQuoteReturn
DATA_LOCATION_8 = MESSAGE_BODY

DATA_MAPTO_9    = EDE_DATA.0.NAME
DATA_REL_PATH_9 = QuoteReturn
DATA_LOCATION_9 = CONSTANT

DATA_MAPTO_10    = EDE_DATA.0.TYPE
DATA_REL_PATH_10 = string
DATA_LOCATION_10 = CONSTANT
```

The following code illustrates the extendedDataElement element fragment of the generated Common Base Event.

```
<extendedDataElements name="QuoteReturn" type="string">
        <values>1,725.00</values>
</extendedDataElements>
```

Please note that if you plan to feed the events to IBM WebSphere Business Monitor, the events generated by the Event Publisher Sub-flow must contain the extension name element for the monitor model to correlate the events and generate business performance-relevant data.

Here is an example of a mapping entry:

```
DATA_MAPTO_8    = EXTENSION_NAME
DATA_REL_PATH_8 = WBI.MonitoringEvent.QUOTE_REQUEST_EVENT
DATA_LOCATION_8 = CONSTANT
```

6.2 Mapping group data to extended data element

Common Base Events support a nested data structure called Extended Data Elements. Extended Data Elements might contain any number of levels. Each level might contain any number of children Extended Data Elements. To simplify the mapping entries, group mapping is supported. The following examples illustrate how to map group data to Extended Data Elements. The following code sample shows the source tree:

```
<MyComplicatedTree>
    <CustomerProfile>
            <PersonalInformation>
                    <Name>Jogvan Oyvind</Name>
                    <Age><25</Age>
            </PersonalInformation>
            <DriverHistory>
                    <NumberOfAccidents>3<NumberOfAccidents>
                    <LicenseStatus>
                            <Level>G</Level>
                            <Expiry>Dec2020</Expiry>
                    </LicenseStatus>
            </DriverHistory>
    </CustomerProfile>
</MyComplicatedTree>
```

Maps all key-value pairs of PersonalInformation into the first level.

To map all key-value pairs in *MyComplicatedTree/CustomerProfile/PersonalInformation* to Extended Data Elements and generate the following:

```
<extendedDataElements name="Name" type="string">
        <values>Jogvan Oyvind</values>
</extendedDataElements>
<extendedDataElements name="Age" type="string">
        <values>25</values>
</extendedDataElements>
```

Mapping entries like the following is required:

```
DATA_MAPTO_8      = EDE_DATA.*
DATA_REL_PATH_8  = MyComplicatedTree/CustomerProfile/PersonalInformation
DATA_LOCATION_8  = GLOBAL_ENVI
```

Maps all key-value pairs of PersonalInformation into the second level

To map all key-value pairs in *MyComplicatedTree/CustomerProfile/PersonalInformation* into the second level of Extended Data Element and generate the following:

```
<extendedDataElements name=”MyPersonalInformation” type="noValue">
    <children name="Name" type="string">
        <values>Jogvan Oyvind</values>
    </children>
    <children name="Age" type="string">
        <values>25</values>
    </children>
</extendedDataElements>
```

Mapping entries like the following is required:

```
DATA_MAPTO_8      = EDE_DATA.0.EDE_DATA.*
DATA_REL_PATH_8  = MyComplicatedTree/CustomerProfile/PersonalInformation
DATA_LOCATION_8  = GLOBAL_ENVI

DATA_MAPTO_9    = EDE_DATA.0.NAME
DATA_REL_PATH_9 = MyPersonalInformation
DATA_LOCATION_9 = CONSTANT

DATA_MAPTO_10    = EDE_DATA.0.TYPE
DATA_REL_PATH_10 = noValue
DATA_LOCATION_10 = CONSTANT
```

Maps all children of the MyComplicatedTree

To map all children of *MyComplicatedTree* to Extended Data Elements as a tree and generate the following:

```
<extendedDataElements name="AdditionalCustomerData" type="noValue">
    <children name="CustomerProfile" type="noValue">
        <children name="PersonalInformation" type="noValue">
            <children name="Name" type="string">
```

```
                                        <values>Jogvan Oyvind</values>
                                </children>
                                <children name="Age" type="string">
                                        <values>25</values>
                                </children>
                        </children>
                        <children name="DriverHistory" type="noValue">
                                <children name="NumberOfAccidents" type="string">
                                        <values>3</values>
                                </children>
                                <children name="LicenseStatus" type="noValue">
                                        <children name="Level" type="string">
                                                <values>G</values>
                                        </children>
                                        <children name="Expiry" type="string">
                                                <values>Dec2020</values>
                                        </children>
                                </children>
                        </children>
                </children>
        </extendedDataElements>
```

Mapping entries like the following is required:

```
DATA_MAPTO_8    = EDE_DATA.0.EDE_DATA.*
DATA_REL_PATH_8 = MyComplicatedTree
DATA_LOCATION_8 = GLOBAL_ENVI

DATA_MAPTO_9    = EDE_DATA.0.NAME
DATA_REL_PATH_9 = AdditionalCustomerData
DATA_LOCATION_9 = CONSTANT

DATA_MAPTO_10    = EDE_DATA.0.TYPE
DATA_REL_PATH_10 = noValue
DATA_LOCATION_10 = CONSTANT
```

Please note that performance degradation may be resulted if a tree has too many children-level. In that case, it is recommended to cast your source tree into hex first and then map it to CBE as HEX_VALUES using mapping entries like the following:

```
# The postfix 'HEX_VALUE' indicates that the source data should be a hex data
# i.e. MyComplicatedTreeInHex is a hex string stored from global environment

DATA_MAPTO_8    = EDE_DATA.0.HEX_VALUE
DATA_REL_PATH_8 = MyComplicatedTreeInHex
DATA_LOCATION_8 = GLOBAL_ENVI

DATA_MAPTO_9    = EDE_DATA.0.NAME
DATA_REL_PATH_9 = AdditionalCustomerDataAsHex
DATA_LOCATION_9 = CONSTANT

DATA_MAPTO_10    = EDE_DATA.0.TYPE
DATA_REL_PATH_10 = hexBinary
DATA_LOCATION_10 = CONSTANT
```

Please note that you must cast your source tree into hex first before calling the sub-flow; otherwise unexpected behavior will be resulted. The Event Publisher Sub-flow will not perform any conversion for you.

7 Appendix B

7.1 Supporting multiple flow-level properties files

You can capture the same set of data at different points of a parent flow by adding the Event Publisher Sub-flow multiple times. To capture different sets of data at different points of a parent flow, you must add the Event Publisher Sub-flow multiple times and use different flow-level properties files. The current asset supports a single flow-level properties file per parent flow. To use multiple flow-level properties files in a parent flow, code modification is required.

To add the Event Publisher Sub-flow multiple times to a parent flow and use different flow-level properties files:

1. Create a new message flow project. For example: **AnotherSub-flow**.
2. In the Java perspective, copy the **CBESenderFlow.msgflow** file from the **CBESenderFlow** project to **AnotherSub-flow** project.
3. Rename the copied **CBESenderFlow.msgflow** file. For example: **AnotherSenderFlow.msgflow**. There are bound to be problems with the new copy of the sub-flow. The following steps fix these problems.
4. Double-click the **AnotherSenderFlow.msgflow** file to open it. Delete the **CBEMapping** Java compute node from the flow diagram. From the palette, add a new Java compute node, and then reconnect the new node to the rest of the flow.
5. Right-click on the new Java compute node and click **Open Java** to create a new Java compute node project. Name the new project **AnotherSub-flowJava** and name the compute node class **AnotherSenderFlow_JavaCompute**.
6. Right-click **AnotherSub-flowJava** project and click **Properties.**
7. Add the **CBESenderFlowJava** project into the Java build path. Click **OK**.
8. Open the **AnotherSenderFlow_JavaCompute** class. Replace the content of this class with all of the methods and fields from the **com.ibm.sal.audit.subflow.CBESenderFlow_JavaCompute** class.
9. Locate the following code in the **AnotherSenderFlow_JavaCompute** class, **evaluate** method.

```
if (mapper.initialize(assembly, getMessageFlow())) {
        // Maps the date from a data tree to a Common Base Event
        mapper.mapData();
```

   To pass in a different properties file name to the **initialize** method, modify the code to match the following example:

```
if (mapper.initialize(assembly, "someDirectory/someFile.properties")) {
        // Maps a date from a data tree to a Common Base Event
        mapper.mapData();
```

   "*someDirectory/someFile.properties*" represents the location of your specific flow-level properties file. For example "*props/SALInsuranceQuoteFlowRouting2.properties***"**

10. Save the changes.
11. Add the **AnotherSub-flow** project to the project references of the parent flow.
12. Add the **AnotherSenderFlow** sub-flow to the right place of the parent flow.
13. Create a specific flow-level properties file (see Section 3.1.3). Specify a folder name and a file name that match the one you specified in step 9 above.
14. Repeat step 1 to 13 to add the Event Publisher Sub-flow at another place of the parent flow.

8 Appendix C

8.1 Installing the package manually

Experienced users can install this package manually. All of the solution files are available in the SolutionFiles folder in the **IA9V.zip** file.

Before installing the package, you must have a broker configuration manager and a broker domain ready. For more information about creating broker domains, see the IBM WebSphere Message Broker Information Center.

To install the package manually:

1. Create a broker configuration manager and a broker domain.
2. Extract the package IA9V.zip to a temporary directory.
3. All of the solution files are available in the SolutionFiles folder.
4. Set up the run-time component for the Event Publisher Sub-flow
    a. Copy all of the JAR files that are in the **EventPublisherSubflow\lib** folder and **EventPublisherSubflow\PropertiesJar** folder to IBM WebSphere Message Broker shared class folder (e.g. *C:\Documents and Settings\All Users\Application Data\IBM\MQSI\Shared-classes*).
    b. Create a queue called **CBE_OUTPUT.** This holds the generated Common Base Events. If you chose a different queue name, you must modify the Event Publisher Sub-flow. See *Section 3.1.1 - Step 6 for* details.
    c. For the queue manager that you chose to manage the **CBE_OUTPUT** queue with, create a server connection channel called **JAVA.CHANNEL.** The channel name is arbitrary.
    d. Restart the IBM WebSphere Message Broker for the shared Common Base Event JAR files to be effective.
5. Set up the design-time component for the Event Publisher Sub-flow.
    a. Create a folder called *C:\Program Files\IBM\IA9V.*
    b. Copy the project interchange archived file **EventPublisherSubflow\EventPublisherSubflow.zip** and the sub-folder **lib** to *C:\Program Files\IBM\IA9V.* If you chose a different target location, then you must fix the build-path of the Event Publisher Sub-flow. See *Section 3.1.1 – Step 4* for details.
6. Install and set up the Event Publisher Message Driven Bean.
    a. Create MQ resources in IBM WebSphere Process Server. You can choose a different name and JNDI name for the MQ resources.
        - Create a MQ queue connection factory called **JMSEventPublisherCF** with the JNDI name **jms/JMSEventPublisherCF** to connect to the queue manager you used in creating the queue in *step 4.b* above. Specify a transport type of **CLIENT** and enter values for **host**, and **port.** For the **server connection channel**, use the same channel you created in *step 4.c*.
        - Create a MQ queue destination called **JMSEventPublisherQ** with the JNDI name **jms/JMSEventPublisherQ** to connect to the queue you created in *step 4.b*. Enter the values for **host**, and **port.** For **server connection channel**, use the same channel you created in *step 4.c*.
        - Create Message Listener Service - Listener Port called **JMSEventPublisherQP** to listen to the connection factory and queue destination created above.
    b. Deploy the Event Publisher Message Driven Bean, **EventPublisherMessageDrivenBean\CEILoggingEAR.ear** to IBM WebSphere Process Server.

   c. Restart the server.
7. Install and set up the insurance quote sample
   a. Deploy the sample insurance quote message flow, InsuranceQuoteSample\InsQMsgFlow.bar to IBM WebSphere Message Broker.
   b. Deploy the sample insurance quote client and provider applications, the following EAR files, to IBM WebSphere Process Server
      - **InsuranceQuoteSample\InsuranceQuoteApps\InsQSampleC.ear**
      - **InsuranceQuoteSample\InsuranceQuoteApps\InsQSampleP_BC.ear**
      - **InsuranceQuoteSample\InsuranceQuoteApps\InsQSampleP_ON.ear**
      - **InsuranceQuoteSample\InsuranceQuoteApps\InsQSampleP_QC.ear**
   c. Start the applications.
8. Include source code of the insurance quote sample
   a. Copy the project interchange archived file **InsuranceQuoteSample\InsQMsgFlow.zip** to *C:\Program Files\IBM\IA9V* which was created in *step 5.a*.

END OF DOCUMENT