A practical approach to integrating JMS and IBM Business Process Manager

Sharad Chandra  
Sateesh Balakrishnan  

January 28, 2013  
(First published July 10, 2012)

In this article, you'll learn how to implement a Java™ Message Service (JMS) initiated Undercover Agent (UCA) with IBM Business Process Manager V7.5 to enable sending of JMS messages directly to the Business Process Manager Event Manager without any custom configuration. This approach provides an elegant way to implement an asynchronous method of invoking a business process in Process Center.

Introduction

Java™ Message Service (JMS) is a messaging system that provides for asynchronous programming styles for Java applications and can also be used to integrate heterogenous platforms. For example, to integrate a third-party application with IBM Business Process Manager V7.5, the normal implementation methodology would be to create a JMS provider (service integration bus, JMS queue connection factory, queue, and so on) in the application server and then create a JMS integration service in Business Process Manager to connect to the configured provider to send or retrieve the message. This article walks you through an integration scenario with Business Process Manager V7.5 using JMS by exploiting the existing messaging infrastructure and the Event Manager of the process server. Although this is a relatively simple implementation pattern, it provides a foundation for developing more complex solutions.

The Event Manager in the Process Server responds to external events. The Event Manager queues incomming messages and triggers the appropriate UCA. Figure 1 shows how the Event Manager receives and processes incomming requests. The Event Manager uses the UCA to locate the service that is associated with the event. So, if you want to invoke a service when an incoming message event is received or if you want to invoke a service as the result of an event that occurs on a regular schedule, you should create a UCA.
The download provided with the article contains two code samples, one for IBM BPM V7.5 and one for BPM V8.

### The role of the UCA

The UCA creates a bridge between external systems and business process definitions (BPDs). You can manage inbound and outbound communication with external systems using UCAs. The UCA listens for events from external systems through a message or specific schedule. If you want to post a message to the JMS listener, the Event Manager has a defined XML message structure that it must receive from an external system. The UCA is responsible for firing either a Start Message Event or an Intermediate Message Event that is contained within a BPD.

The following queues are associated with a UCA:

- Async queue: Allows Event Manager jobs to run at the same time.
- SYNC_QUEUE_1: Forces one job to finish before the next job can start. By default, three synchronous queues are available.
- SYNC_QUEUE_2: Forces one job to finish before the next job can start. By default, three synchronous queues are available.
- SYNC_QUEUE_3: Forces one job to finish before the next job can start.

By default, three synchronous queues are available. All of these queues gets initialized at the time of server start up.

### Step 1. Create a process

The process application used in this article is a simple human service implemented as Coach, as shown in Figure 2, that displays the data received through a message event that is configured to interact with UCA.
Figure 2. Coach to display received data

Figure 3 shows the input data that process receives and publishes in the Coach.

Figure 3. Input data for the process

Step 2. Create an event handler service

An event handler service is nothing but a General System service, which is the heart of a UCA. This service receives the data from external stimulus and sends it to a BPD process through the UCA. You need to define and map output data to input data explicitly for this service, otherwise the output data would be null. The data defined for this service should be same as the input data required by the BPD process. To define the service, complete the following steps:

1. Select **General System Service**, as shown in Figure 4, and name the service ComplexUCAService.
2. Define the input data and output data required by the BPD process using the **Variable** tab, as shown in Figure 5.

**Figure 5. Create Input and Output parameters for the UCA service**

3. Add a server script and wire it to the input and output. Call the script `Input_OutputMapping`, and map the input data explicitly to the output variable in the **Implementation** property of the component as shown in Figure 6.
Step 3. Create the UCA

You must create the UCA before it can be referenced. To create a UCA and associate it with the General System service, complete the following steps:

1. In the Designer view, select the plus sign (+) next to **Implementation**, then select **Undercover Agent**, as shown in Figure 7.

2. In the New Undercover Agent dialog, specify the following information, as shown in Figure 6, and then click **Finish**.
   - **Name**: Specify a name for the new UCA.
   - **Schedule Type**: Select **On Event** from the list.
   - **Attached Service**: Click **Select** to select the service to invoke when the event is received. For this case, select **ComplexUCAService**.
3. Once the UCA is created, a dialog displays in which you can configure UCA properties, as shown in Figure 9. Here you need to associate the UCA with the new service you just created. Make sure to check Enabled to enable the execution of the service when an event arrives. If the checkbox is not checked, the Event Manager does not run the UCA when an external message is received. Note that the Event Manager monitor may show that the Event Manager has executed the UCA, but if Enabled is not checked, the execution does not occur.

In the Parameter Mapping section, check Use Default to use the default value of the input variable in the attached service. If the input variable of the attached service does not have a default value, this checkbox is disabled. In most cases, the required values are included in the incoming message event and no action is required here.
Step 4. Send a message to the Event Manager

To invoke a BPD process by sending application data through the UCA, you need to encapsulate data in a predefined XML structure with `eventmsg` as the root, as shown in the following listing:

```xml
String data = "<eventmsg>
<event processApp="JMSTA" snapshot="JMSTASNAPSHOT"
ucaname="complexBOUCA"></event>
<parameters>
  <parameter>
    <key>ucacomplexInput</key>
    <value>
      <customerID>"+customerID+"</customerID>
      <name>"+name+"</name>
      <address>"+address+"</address>
      <surname>"+surname+"</surname>
      <city>"+city+"</city>
      <state>"+state+"</state>
      <country>"+country+"</country>
    </value>
  </parameter>
</parameters>
</eventmsg>
```

Notice that you need the following information available to include in the XML header:

- **snapshot**: Process snapshot name; created through Process Center.
- **ucaname**: UCA name; available in the UCA editor in Process Center.
- **processApp**: The process name; this acronym is provided when you create a new process app.
- **event**: The event name; in this case, `complexBOUCA`.
- **key**: The input object name defined in the UCA service.

In our scenario a servlet client is used to send a message to the Event Manager. The servlet dynamically formats the message in the prescribed format. For the message to hit the UCA service, you need to look up the event queue in the underlying WebSphere Application Server. The JNDI referring to this event queue is `jms/eventqueue`, as shown the following example:

```java
Queue ucaqueue=(Queue)ctx.lookup("jms/eventqueue");
```
For more information on message structure, refer to **Understanding message structure** in the Business Process Manager Information Center.

**Step 5. Define the message event and associate it with the UCA**

A message event is the actual component that collaborates with the UCA to receive the external stimulus to invoke the BPD process. You need to wire this event to a process task by doing the following:

1. Create a BPD process in the Process Designer. Drag and drop the Start Message Event onto the palette and connect it to the Submit Request activity using a sequence line, as shown in Figure 10.

   **Figure 10. Create a message event**

2. Associate the message event with the **complexBOUCA** UCA you defined previously, as shown in Figure 11.
Step 6. Test the integration solution

Before you can test the solution, you need to expose the process to the AllUsers group, as shown in Figure 12. Otherwise, the process will not be invoked and visible in Process Portal.

Figure 12. Expose the process to AllUsers

To test the overall solution, we created a simple JSP-based client application which is deployed in WebSphere Application Server. Using this JSP client, provided for download with this articleCDR: Sharad, is this reference still okay?? and shown in Figure 13, you can create a request that is submitted to the servlet. The servlet posts the message to the queue where it is consumed by the Event Manager.
Once you have submitted the message via the JSP client, you can log into the Process Portal as admin. In the inbox, you should see the new process instance that was created as a result of the message, as shown in Figure 14. If you click on the green "Run" icon in the process instance and run the process, you will see the data received from the client, also shown in Figure 14.

Figure 14. Results in Process Portal

Step 7. Verify the results in the JVM log
Once the UCA is triggered successfully, you can verify the success of the invocation through Business Process Manager JVM log as shown in Figure 15.
Conclusion

This article described how to initiate a process by sending a JMS message to the Event Manager of IBM Business Process Manager V7.5. It walked you through the steps to implement the Event Manager and the UCA, and described the runtime relationships between them. You learned how to integrate JMS with Business Process Manager using an associated BPD and the default messaging infrastructure of the Process Center.

Acknowledgements

The authors would like to thank Scott Simmons and Harsha Bhushana for their review of this article.
## Downloadable resources

<table>
<thead>
<tr>
<th>Description</th>
<th>Name</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample scenario</td>
<td>JMSUCATEST.zip</td>
<td>9KB</td>
</tr>
<tr>
<td>Sample code for IBM BPM V7.5 and V8</td>
<td>SampleCode.zip</td>
<td>794B</td>
</tr>
</tbody>
</table>
Related topics

- IBM Business Process Manager V7.5 Information Center
- Understanding message structure (in the IBM Business Process Manager V7.5 Information Center)
- developerWorks BPM zone: Get the latest technical resources on IBM BPM solutions, including downloads, demos, articles, tutorials, events, webcasts, and more.

© Copyright IBM Corporation 2012, 2013
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)