Bonita for business process management, Part 1: Configure a simple workflow

Leverage an open source Java-based tool to model and manage business processes

Bilal Siddiqui

Open source Java™-based tools for business process management (BPM) are coming into their own. In this two-part article, Bilal Siddiqui introduces BPM concepts and shows the features of Bonita Open Solution — a BPM engine that implements the Business Process Modeling Notation (BPMN) standard. In Part 1, you'll learn how various BPMN elements work and start configuring an example business-process workflow with Bonita. In Part 2, you'll complete the remaining configuration tasks to implement the workflow.

View more content in this series

A business process occurs when people interact in a coordinated manner to achieve a common business goal. Work flows from one person to the next, and each individual performs his or her own task or business role. For example, consider a simple workflow for checking a customer into a hotel. The receptionist books the room for the arriving guest. The room-booking information travels to the housekeeping department and the hotel's accountants. The housekeeping department ensures that the customer finds the room ready, and the accountants keep a record of the customer's payments.

Management of a business process is management of the flow of work. Business process management (BPM) tools allow you to express workflows in a way that lets computer systems understand them and act accordingly. The old way of doing this is to write code in some programming language. The BPM way is to do it administratively through configuration without writing a single line of code, using BPM software. Business Process Modeling Notation (BPMN) is a standard that lets you express a workflow's business requirements graphically. You first use a BPMN workflow editor to author the BPMN representation of your business process. Then you host the BPMN representation on a BPMN workflow engine, which handles the actual flow of work.

Some excellent Java-based open source BPM solutions are available. In this two-part article, you'll learn how to use one of them — Bonita Open Solution — to configure a workflow for a hotel's
room-booking business process. Bonita includes a graphical BPMN editor as well as a workflow engine.

Tasks in configuring a workflow

Configuring a workflow requires four tasks:

1. Author the workflow using BPMN.
2. Design the forms used for human interaction during the workflow's execution.
3. Specify which external applications are to be used in your workflow. You will almost always need external applications to fulfill your workflow requirements.
4. Specify who performs which activity of the workflow by defining the business roles of all personnel (or actors) taking part in it. Business roles determine who has access to which activity. Deciding all this is part of an access-control policy.

Here in Part 1, I'll demonstrate the first task, using Bonita to author the room-booking workflow. In Part 2, I'll show how to design two forms for the workflow and demonstrate working with two popular open source applications frequently used in workflows: the PostgreSQL database and the JasperReports reporting engine. Defining access-control policies is beyond this article's scope; see Related topics for a link to an article on that topic.

BPMN representation of a simple workflow

To introduce you to BPMN, I'll use the room-booking process at a hotel's front desk as an example. The process requirement in its simplest form is that a customer arrives at the desk and asks for a room. The front-desk clerk checks the availability of rooms and, if a room is available, books it. (Note that for the sake of simplicity, I am not considering the complete workflow, mentioned in the introduction, from the hotel's front desk to the housekeeping department and accountants.)

BPMN representation of this simple process (which I drew using Bonita's workflow editor) is shown in Figure 1:

Figure 1. Using BPMN to represent a simple workflow to book a hotel room

The thin green circle at the left-most position in Figure 1 is the start element. It represents the start of a workflow as defined by BPMN.

The arrow to the right of the start element represents a transition. Various transitions are possible, as you'll learn in this article. This one is a simple transition that only points to the workflow's first activity.
Activities of a workflow

To the right of the first transition in Figure 1 is a rectangle with rounded corners. Figure 1 contains three such rectangles. They represent workflow activities or tasks, which can be automatic or performed by a user. The left-most rectangle — named EnterRoomInformation — contains a symbol of a human, indicating that it’s a user activity. In this activity, the receptionist uses a form to enter information about the type of room the customer requires.

BPM and ERP compared

Enterprise resource planning (ERP) tools (see Related topics) provide another way to manage your business workflows. When configuring an ERP implementation, you perform similar tasks to those in BPM (workflow authoring, form designing, linking to external applications, and defining access-control policies). As ERP tools have evolved over time, they’ve competed with one another to offer richer workflow features. All ERP tools offer preconfigured workflows and some let you choose among workflow types — categories of workflows commonly required in managing business processes. You can use workflow types to configure applications quickly according to your business requirements. The result of the evolution of ERP tools is that they are much less standardized than BPM tools. BPM is a more organized and more standardized solution to the same problem that ERP tools have been trying to solve.

The middle rectangle is named CheckRoomAvailability. Its plug-and-wire symbol shows that it is an automatic activity, which means it does something at the back end. Several types of automatic things can be done — for example, a database operation to check whether the required type of room is available.

The third (right-most) rectangle, named BookRoom, has both a human and a plug symbol, which means a user performs part of this activity and the rest is done by some automatic process at the back end. The human part of the BookRoom activity is for the receptionist to provide room-booking information. The rest is automatic (a database transaction to save room-booking information).

In summary, the first activity simply gathers data about the room required, and the second checks whether the required type of room is available. The third activity does everything to book the room.

Simple and conditional transitions

Note the simple transition (an arrow) in Figure 1 pointing from the EnterRoomInformation activity to the CheckRoomAvailability activity. This simple transition means that after the receptionist gathers room-query data, the next step is always to check the availability of the required type of room.

The next transition — from the CheckRoomAvailability activity to the BookRoom activity — is not a simple transition. It depends on the outcome of the CheckRoomAvailability activity — that is, whether the required type of room is available. If it is available, the clerk may want to book it for the customer, so the BookRoom activity takes place. If the required room type is unavailable, the clerk may want to check the availability of another type of room, and so control should be transferred back to the EnterRoomInformation activity.

So, here you have to make a decision about which way to go. Whenever you have to make this decision, the path of execution splits. This split is represented by a diamond (a gate element
or *gateway* in BPMN terminology) with a cross inside it. You can see in Figure 1 that the gate element connects the output of the CheckRoomAvailability activity with the two possible paths of execution.

Three lines with arrows connect to the gate. Each line represents a transition:

- The first transition is the input to the gate, which originates from the CheckRoomAvailability activity and terminates at the gate. This is a simple unconditional transition.
- The second transition is from the gate to the EnterRoomInformation activity. This transition represents a case when a room of the specified type is not available. It is a conditional transition.
- The third transition from the gate — another conditional transition — points to the BookRoom activity, in case the required type of room is available.

Note that the gate element is nothing but a point where transitions meet. Each transition that enters the gate or leaves it knows its own properties (that is, whether it is simple or conditional).

Now you've seen how the simple process of booking a room is represented by BPMN. The next section demonstrates how to configure this workflow in Bonita. Download, install, and launch Bonita now if you want to follow along with this article's hands-on exercise (see Related topics).

**Configuring a simple workflow in Bonita**

Bonita's graphical tool is simple to use, allowing drag and drop from a palette into an editor window called a *whiteboard*. After dropping components onto the whiteboard, you can configure each component individually according to business-process requirements.

Figure 2 shows Bonita's welcome page, which contains quick links for common tasks such as creating a new process or opening an existing one:
You can click on the **New** button to start configuring your room-booking workflow. A new process named **MyProcessDiagram (Version 1.0)** opens, as shown in Figure 3:

**Figure 3. A new process**

You can see in **Figure 3** that the Bonita screen is divided horizontally into two parts, and each part is divided vertically as well.

The upper-left section contains a palette of BPMN elements that you can use to configure your workflows.
The upper-right portion is a process whiteboard into which you drag and drop BPMN elements from the palette. Whiteboard is the actual design view of the process. As Figure 3 shows, Bonita automatically adds three BPMN elements while creating a new process: start, a transition, and an activity named Step1.

The lower-left portion is an overview window, which shows an overview of the entire workflow you are developing. It is helpful for jumping quickly to a particular portion of your workflow, especially when the workflow definition is large and can't fit into the whiteboard without scrollbars.

The lower-right portion is a details window, which you use to configure individual BPMN elements. It shows the BPMN attributes that you set during configuration. Because no BPMN element is currently selected, the details window in Figure 3 shows attributes of the workflow. For example, you can see that the details window has a tab named Diagram, which contains a Name field whose value is MyProcessDiagram. Change MyProcessDiagram to RoomBookingDiagram and click the Save button on Bonita's main toolbar.

Configuring a user activity in Bonita

Start configuring the room-booking workflow by clicking on the Step1 activity. Its attributes appear in the details window, as shown in Figure 4:

**Figure 4. Attribute details of the Step1 activity**

When you click on the Step1 activity, the context palette — a small subset of BPMN element icons — appears around it, as shown in Figure 5:
Figure 5. The Step1 activity showing its context palette

The context palette lets you conveniently add the next activity or transition to your workflow. You'll use the context palette often in this article's exercise.

You can see from Figure 4 that the details window has several tabs, which you can use to control the behavior of each BPMN element — such as the Step1 activity — that you drag and drop into Bonita's whiteboard. These tabs offer rich features to cover almost any possible requirement of managing business processes. I won't cover all the BPMN features in this article. For the simple hotel room booking use case, you need only the General, Data, Connectors, and Forms tabs of the details window.

The General tab, as shown in Figure 4, lets you specify the name, description, type, and priority for your activity. Replace Step1 as the name of the activity by typing EnterRoomInformation the Name text field. Notice that EnterRoomInformation is now the name of the activity that appears on the whiteboard.

The Description field allows you to provide documentation about the BPMN elements. Type show a form to fetch the information about the room required by the customer in the Description field.

Next you'll specify the type for this activity. Because the EnterRoomInformation activity requires filling a data-entry form to specify the type of room required, this is a user activity performed by the receptionist who uses the room-booking workflow application. The Activity type field is a drop-down list. The type of EnterRoomInformation activity should be Human, which you'll find selected by default.

Because EnterRoomInformation activity is a user activity, you need to develop a data-entry form for it. In Part 2, you'll design and implement the form, then add it to the activity by using the Forms tab of the details window. In Part 2, you'll also use the Data tab to define variables that hold the data about the room the customer requires.

Adding an automatic activity to your workflow

Next you'll add a transition and the next activity, named CheckRoomAvailability, to your workflow. The CheckRoomAvailability activity receives data from the EnterRoomInformation form, checks the hotel's database (which you will configure in Part 2) for room availability, and then sets a flag to show whether the room is available.

Click on the EnterRoomInformation activity to activate its context palette. Drag the step element icon (a small box) from the top-right corner of the context palette and drop it next to the EnterRoomInformation activity. As shown in Figure 6, a transition and another Step1 element are added to the workflow.
The details window of the newly created Step1 element shows the name, description, and type fields for the activity. Enter CheckRoomAvailability as the name of the activity. In the Description field, type An automatic process to consult the database for the availability of required type of room. Select Automatic from the Activity type drop-down list. Figure 7 shows the CheckRoomAvailability activity as you’ve just configured it:

Adding a room-availability flag to your workflow

If the required type of room is available, the CheckRoomAvailability activity sets a flag named isRequiredRoomTypeAvailable. The next transition in the workflow depends on whether the isRequiredRoomTypeAvailable flag is set or not, so you need to define the isRequiredRoomTypeAvailable flag as a variable in Bonita.

Bonita allows you to define variables for each individual activity as well as for the complete workflow. You define an activity-level variable when the variable is required only within a particular activity. If the use of that variable spans multiple activities, you define the variable at the workflow level so that any element can access it.

In case of the room-booking workflow, the isRequiredRoomTypeAvailable flag needs to be a workflow-level variable, because multiple elements access the flag. The CheckRoomAvailability activity sets the flag, and the next transition checks the flag’s status.

To define a variable at workflow level, click on the rectangle wrapping the entire workflow definition, highlighted in Figure 8:
Figure 8. Details window for the room-booking workflow

Notice that the details window now shows attributes of the complete room-booking workflow.

Click on the Data tab in the details window and click the Add button. The Add a new variable dialog opens, as shown in Figure 9:

Figure 9. Adding a new variable to your workflow

Enter `isRequiredRoomTypeAvailable` in the Name text field. In the Description field, type a workflow-level flag to show whether the required type of room is available or not. Select `Boolean` from the Data type drop-down list, and choose `false` as the default value of the flag. Click the Finish button at the bottom. You can see that `isRequiredRoomTypeAvailable` variable has been added to the Data tab, as shown below in Figure 10:
You have just defined the CheckRoomAvailability activity along with a workflow-level flag named `isRequiredRoomTypeAvailable`. However, you have not configured the actual database and SQL statements that will read the room availability data and set (or reset) the `isRequiredRoomTypeAvailable` flag. I will demonstrate doing all this in Part 2. For now, I'll show how to configure the next dynamic transition, which requires decision making depending on whether the room is available.

Dynamic transitions with decision making

Click on the CheckRoomAvailability activity to activate its context palette. Drag the diamond with a plus sign from the context palette and drop it just to the right of the CheckRoomAvailability activity to place a gate element there, as shown in Figure 11:

**Figure 11. A gate element for decision making in a BPM process**

Gate elements can be of two types:

- XOR gate: If the gate has a cross sign inside it, it's used to split one path into several paths.
- AND gate: If the gate has a plus sign inside it, it combines two paths.

You know from the Simple and conditional transitions section that you need a gate that splits one path into several paths. In order to set the correct type of gate, click on the gate element in the whiteboard, then choose XOR as the Gate type property in the General tab, as shown in Figure 12:
Figure 12. Choosing a XOR gate to split one path into several

![Image of XOR gate configuration]

Now click and drag the step element icon from the gate element's context palette to the gate's right. A new activity named Step1 is placed to the right of the gate. Change the name of the new activity to BookRoom and its type to Human from the General tab in the details window. Your gate is now configured, and the process looks like Figure 13:

Figure 13. The BookRoom activity placed next to the gate

![Image of process flow with BookRoom activity]

Business roles in Bonita

On the left corner of the BookRoom element is a symbol representing a human, and on the right corner is a cross (both visible in Figure 13). The cross tells that you have not specified who will do this activity. Click on the Actors tab and then click the Choose button. An Assign actors dialog opens, as shown in Figure 14:
Figure 14. The Assign actors dialog

The Assign actors dialog asks you to select the groups of actors who can perform the BookRoom activity. Bonita allows you to define groups of actors according to the business roles required in a workflow. Discussing business roles is beyond the scope of this article. You'll simply use a default actor group named Initiator, already defined in Bonita. The Initiator group consists of anyone who initiates the workflow.

Select Initiator from the text box and click the Finish button at the bottom of the window. The Initiator is added to the list in the Actors tab, as shown in the Figure 15:

Figure 15. The default Initiator actor added to the list in the Actor tab

The BookRoom activity also requires a user-interaction form that asks the hotel's front-desk clerk to enter customer information. You'll design the forms required for the workflow in Part 2. Also notice that for the moment, BookRoom is a user activity. In Part 2, you'll configure the automatic part of this activity.

Adding a condition to a transition

Now you'll add the room-availability condition to the transition from the gate to the BookRoom activity. You need to configure the workflow in such a way that the BookRoom activity takes place only when the required type of room is available.
Click on the transition element from the gate to the BookRoom activity. Details of the transition element appear in the details window, as shown in Figure 16:

**Figure 16. Details window showing attributes of a transition**

Details of the transition element contain just one tab, named General, as shown in Figure 16. The General tab contains a field named Condition, which specifies the condition under which the transition will take place. Enter isRequiredRoomTypeAvailable in the Condition text field. Recall from Adding a room-availability flag to your workflow that you declared the isRequiredRoomTypeAvailable flag as a workflow-level variable. If the room is available, the CheckRoomAvailability activity sets the isRequiredRoomTypeAvailable flag to true. This condition simply enables the transition to the BookRoom activity if the isRequiredRoomTypeAvailable variable is found to be true (meaning the flag is set).

As you enter a condition for the transition, notice that a small diamond appears at the gate side of the transition element. The diamond shows that the transition is conditional.

As you might have guessed, you also must set a condition for the other transition. Click on the gate element in the process whiteboard to activate its context palette. Drag the arrow (the transition icon) from the bottom of the context palette to the EnterRoomInformation activity. This draws a transition element from the gate to the EnterRoomInformation activity, as shown in Figure 17:
Figure 17. Transition from the gate to the EnterRoomInformation activity in case the required room type is unavailable

Now, as shown in Figure 17, simply set `isRequiredRoomTypeAvailable` as the condition for this transition. The work will flow back to the EnterRoomInformation activity if the required type room is unavailable.

Configuring the end of your workflow

The final workflow-configuration step is to add an end element to your process by dragging the circle item from BookRoom’s context palette toward the right. An event-selector palette appears, as shown in Figure 18:

Figure 18. The event-selector palette

The event-selector palette contains symbols for many types of events defined by BPMN. You just need the end element (a red circle, shown encircled in Figure 18), which you can find at the top-right corner of the event-selector palette. Select the end element from the events palette. Your workflow now contains an end element, as shown in Figure 19:
You have configured the room-booking workflow, except it has no forms and no database. Save the work you've done so far by clicking **Save** in Bonita's main menu. This saves your workflow information in the RoomBookingDiagram.proc file. The RoomBookingDiagram.proc file is also available for download.

**Conclusion**

ERP features like preconfigured workflows and popular workflow types are useful, but they come with a price: such features are product-specific. I've found that teaching IT professionals how to use product-specific features to solve everyday workflow problems is the hardest part of any ERP project.

For this reason, I think BPM is the wave of the future. Although standardization is not yet complete, it is progressing. I am confident that within a few years, BPM will cover all workflow features, common or uncommon, so that standardized BPM training will become part of the IT curriculum, and product-specific ERP training will no longer be an issue.

In this article, you've learned a bit of BPMN and how to use it to configure a simple workflow in Bonita. In **Part 2**, I'll cover all that I left undone in Part 1 — designing forms, using variables, connecting to a PostgreSQL database, and generating reports with JasperReports — to launch your room-booking workflow into action.
## Downloadable resources

<table>
<thead>
<tr>
<th>Description</th>
<th>Name</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>BPMN file for the example</td>
<td>j-bpm1.zip</td>
<td>4KB</td>
</tr>
</tbody>
</table>
Related topics

- **Object Management Group/Business Process Management Initiative**: Check out the BPMN specification and other BPMN resources.
- **Bonita Open Solution**: Visit the Bonita website.
- **BPMN 2.0 new features**: Watch this video to see some of the BPMN 2.0 features available in Bonita.
- **ADempiere, Compiere, and OpenBravo**: Learn more about these Java-based open source ERP products.
- **Bonita Open Solution**: Download Bonita. The article uses version 5.2.2.
- **Start developing** with product trials, free downloads, and IBM Bluemix services.

© Copyright IBM Corporation 2010

Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)