Implementing a Business Process in Oracle BPM 11g: Narration Script

Implementing a Business Process in Oracle BPM 11*g*

Hello, and welcome to this online, self-paced course entitled *Implementing a Business Process in Oracle BPM 11g*. My name is Rosie Piller, and I will be your tour guide for the next 40 minutes of interactive lectures, product demonstrations, and review sessions. As the title suggests, this course aims to familiarize you with what's involved in implementing a business process using Oracle Business Process Management 11*g*, also known as Oracle BPM 11*g*.

Using the Player

Before we begin, take a look at some of the features of this Flash-based course player. If you've attended a similar self-paced course in the past, then feel free to skip this slide.
On the left of your screen, you will find a hierarchical outline. This format enables and even encourages you to go at your own pace, which means you are free to skip over topics you already feel confident about, or jump right to a feature that really interests you, or go back and review topics that were already covered. Simply click a slide title in the outline to display its contents. However, note that by default we will automatically walk you through the slides without requiring you to use the outline.
Standard Flash player controls are found at the bottom of the player, including Pause, Previous, and Next buttons. There is also an interactive progress bar to fast forward or rewind the current slide. Interactive slides may have additional controls and buttons along with instructions on how to use them.
The audio narration script for this course is available from the Attachments button.
This course will now pause, so feel free to take some time and explore the interface. Then when you're ready to continue, click the Next button below or alternatively click The BPM Life Cycle in the outline on the left.

The BPM Life Cycle: Introduction

Business Process Management is a strategy for managing and improving the performance of a business through continuous optimization of business processes in a closed-loop cycle of modeling, implementation, execution, and measurement. Oracle follows a methodology that it developed over many years of working with a wide variety of companies. Here is the typical life cycle Oracle uses for business process management. Click each phase of the life cycle to hear more details. If you're familiar with this life cycle, feel free to advance to the next slide.

The BPM Life Cycle: Planning, Strategy, Analysis, Design

Just as with any large project, good planning is important to the success of a BPM initiative. A company with a high level of BPM maturity goes beyond the departmental level and takes a comprehensive view of the entire enterprise—namely, its goals, operations, processes, and IT systems—so it can align BPM projects with business objectives. Coming out of the planning and strategy phase, you'll have a BPM steering committee, a BPM scope document, some BPM process candidates, and a BPM repository that will be used to hold requirements, data models, and various project artifacts. BPM process candidates should be high-value processes which are amenable to automation and have a high benefit-to-risk ratio.
The Enterprise Architect will then analyze the technical aspects of the BPM process candidates and create a BPM Road Map. The BPM Road Map describes the current state, creates the future

vision, identifies the gaps between the two, and defines a road map to get from the current state to the desired state.

The BPM Life Cycle: Model and Simulate

Once the Enterprise Architects have completed their work, Process Analysts model the flow of a business process, document its steps, and define process metrics based on Key Performance Indicators and Service Level Agreements identified by business users. Models are simply a way for Process Analysts to document processes in a structured way. A model is a “picture” of the process, if you will, and as the saying goes, a picture is worth a thousand words.
A very important and sometimes overlooked part of modeling is simulation. This involves testing the process with real world assumptions regarding staffing levels, system performance, cost, throughput, and so on. It’s worth taking the time to do simulations, because it helps to run what-if experiments to see what the impact would be of, say, reducing or increasing the number of people assigned to an activity, or a sudden increase in the number of items flowing through the process.

The BPM Life Cycle: Implement and Deploy

After the Process Analysts have modeled the process, Developers need to *implement* the process. In this step, Developers refine the process model and make technical configurations so it is executable. For example, for each activity in the process, they define the necessary technical components to ensure the appropriate participants are notified when they have a task to perform. They create the forms end users will see when they participate in the process. And they integrate the process with back-end databases and other systems.
After the implementation work has been done and thoroughly tested, Administrators perform some configuration tasks, and you are ready to deploy the process to run time. This is when you go live with the process, making it available to end users.

The BPM Life Cycle: Execute

Once the process is deployed, it is used within the business, and end users start participating. For example, a Sales Rep can enter a quote, and according to the process flow, participants are automatically notified when they need to perform a task. Some of the tasks may be performed by the system, without any human interaction, according to how the process is modeled and implemented.

The BPM Life Cycle: Manage

Now that your project is up and running, Administrators need to *manage* the deployed application, to ensure it is running properly and to resolve any issues that occur.

The BPM Life Cycle: Monitor and Analyze

You've come a long way, but your job is not done! Process Owners will want to monitor the running process to see how it is performing. With Oracle BAM, you can even set up alerts on abnormal business conditions, or configure automated actions when certain thresholds are passed.

The BPM Life Cycle: Collaborate

A key factor in the BPM life cycle is collaboration. For example, there might be several Process Analysts collaborating on a single BPM project. And Developers may need to interact with

Process Analysts in order to tweak the process model. Process Owners may want to review the process model before it goes live. And so on.
The Oracle BPM Suite facilitates collaboration by providing tools that allow all parties to share the same process model.

The BPM Life Cycle: Continuous Improvement

As we mentioned earlier, Business Process Management is a strategy for managing and improving the performance of a business through continuous optimization of business processes in a closed-loop cycle of modeling, implementation, execution, and measurement. So this is an iterative process. Actual performance data from the monitoring phase can be fed back into the model in preparation for another round of performance improvement. Companies that take the time to do this and manage the resulting changes effectively are poised to excel.
Oracle BPM supports the whole BPM process life cycle. It provides tools used by all participants in each stage of the BPM life cycle.

The BPM Life Cycle: Implement and Deploy

In this course we focus on how Developers implement the process, test it to ensure it is working properly, and then deploy the process to BPM Run Time.

Course Road Map

Here are the topics we plan to cover in this course. Let’s start with a brief overview.

Sample Sales Quote Application

We’ll use as an example a simple Sales Quote process, in which a Sales Representative enters a quote into the system, and the quote goes through several stages of approval before the contracts are signed. The process model you see here reflects what a Process Analyst might hand over to a Developer for implementation: it shows the major tasks and indicates the branching process instances can take after the Sales Representative enters a quote.
I’ll pause the course so you can examine the process model in more detail. When you’re ready to move on, click the Next button.

Why Implement?

So the question is, how do you get from a process model to a running process, which automatically routes tasks to assigned participants, brings up the right forms, pre-populates those forms if necessary, applies business rules you may have defined, makes the necessary calls to services, stores data, and so on.
Implementation is a big part of that process, as Developers integrate each element of the process with services and back-end systems.

Implementation: Steps

At a high level, here are the tasks a Developer performs in order to implement a process model. First, you decide what data objects the process will need, and you declare variables for these data objects.

Then, for each activity in the process, you define the appropriate service component: a business rule to implement a business rule activity, a human task to implement a user task, and so on. You bind these service components to their activities, and you map data into and out of the activity. For gateways in the process that have conditional outgoing transitions, you define expressions that determine which branch a particular instance will take.
You need to create the necessary approval groups, if any, and you can map BPM Studio roles to LDAP users and groups.
There are a number of other enhancements you might need to make, for example to handle errors and inter-process communication, and then you are ready to test and deploy the project.
In fact steps 2 through 5 can be done in any order, but we'll present them as you see here.

Main Tool: BPM Studio

The main tool you'll be using is BPM Studio, and you are in fact accessing the same process model as the Process Analyst, which makes collaboration easy.
Your focus will be different from that of the Process Analyst, however. For example, you'll be using the Business Catalog and Structure panel heavily, along with other views not shown here.

Implementation: First Step

So let's dive right into the first topic: creating data objects to store process and instance data.

What Are Data Objects?

Data objects are variables that are defined during the implementation of a process. At run time, they are used by process instances to store specific information that is needed during the execution of a task. The information is also evaluated by gateways to determine branching.
Data objects can be either at the process level (available only to the process) or at the project level (in which case each process in the project has its own copy of the data object).
So for example, when the Sales Rep enters a quote, there's a data object called *quote* that includes all the information he or she supplies. The Determine Approvals business rule task takes *quote* in as input, applies business logic, and outputs values that determine branching in the gateway below it and also what approvals are needed for the Approve Deal task later in the process. This conditional transition evaluates one of those values. Business Practices Review takes in the quote and outputs whether Business Practices Review passed or failed. All of this is handled using data objects.
So when implementing a process, you need to ask yourself what information you will need to make available to activities, gateways, rules, and so on.

Sample Data Objects

Here are the data objects required by the RequestQuote process. First, on the left, you see the quote data object—a complex one that stores all the details about a quote. Please pause the recording if you'd like to take a closer look.
In the list in the middle, you see approvalFlow. This too is a complex data object and stores information about whether Business Practices Review is required as well as approval specifications for the Approve Deal task.
Three of the other data objects store the outcomes of three of the approval tasks: Approve Quote, Approve Terms, and Business Practices Review.
The data objects on the right are business indicators that the business users would like measured, in addition to the process metrics that Oracle BPM stores as a matter of course.

Hopefully this gives you a sense of what data objects are, and how they are used.

Implementing Activities

Next on our list is implementing each activity in the process. Each type of activity requires a different kind of implementation, but they have one thing in common—you usually need to map data into and out of the activity.

Activities: Basic Properties Tab

To demystify this somewhat, let’s take a look at the properties for three types of activities—a user task, a business rule task, and a service task. Notice the Basic tab, shown here, is very similar for all three tasks, except for the activity name, of course, and the icon representing the activity.

Activities: Implementation Properties Tab

In contrast, the Implementation tab for each of these activities is very different. Here you see implementation properties for a user task. The Implementation tab for a business rule task has different elements in it, and for a service task, different from either of the first two. Yet all three include a Data Associations section so data can be mapped into and out of the activity.

Course Road Map

The next few sections show you how to implement three sample activity types: business rule tasks, user tasks, and service tasks. Let’s start with implementing business rule tasks.

What Are Business Rules?

Business rules are statements that describe business policies or key business decisions.
For example, a business rule for a car rental company might specify that if the driver’s age is less than 21, deny the application.
Process Analysts might create the Business Rule task and document it, but it’s the Developer’s job to implement it.

What the Determine Approvals Task Does

In the RequestQuote process, Determine Approvals is a business rule task. It serves two important purposes: it determines whether a quote requires Business Practices Review, and it defines the approval chain and list of approvers for the Approve Deal task, which occurs later in the process flow.

Business Rules: What Developers and Business Users Can Do

Developers use BPM Studio to implement business rules and deploy the projects to BPM Run Time. At run time, control goes to the Business Rules engine, which executes the business rule and then returns control to the BPMN engine.
Also at run time, Business Users can use Business Process Composer to update rules of deployed projects. As this diagram shows, both tools access the same Business Rules Asset Catalog.
Think about what that means. If a car rental company decides to lower the acceptable age to 20 instead of 21, a Business User could make this change without programmer assistance and without interrupting the deployed business processes. They can change values and commit those

changes to the rule dictionary that is already running, thus altering behavior without having to touch the rule logic.

Implementing Business Rule Activities

So what's involved in implementing a Business Rule activity? Well, you need to define a business rule (which is essentially a decision function), bind that business rule to the activity, and then map the data into and out of the activity.

What Approvals Are Required?

The Determine Approvals rule is fairly sophisticated. Recall that the first thing it determines is whether a process instance goes through Business Practices Review. That part of the rule is fairly straightforward: if the requested discount is greater than the pre-approved discount, which is 30%, Business Practices Review is required; otherwise, it is not. Based on this rule, Determine Approvals sets the value the gateway uses to determine whether a given instance should be routed through Business Practices Review. Who performs Business Practices Review? Those who have the swimlane role, namely, the BusinessPractices role.
Determine Approvals also defines the approval chain and list of approvers for the Approve Deal task, and as you can see, this is far more complex. Both discount and revenue play into that decision. The business rule logic has to determine whether tier3, tier2, and tier1 approvals are required, and then, exactly what the list of participants should be. The three tiers in the Sales Quote example demonstrate three different approval patterns: a serial management chain approval pattern, a pattern in which the members of a group perform approval in parallel and the members of that group are determined dynamically by the business rule, and a specifically assigned approval group defined in Business Process Workspace.
Note that the list of approvers for the Approve Deal task has nothing to do with the swimlane the task is in. Approve Deal is what's known as a Complex user task, and the participant list is determined by the Approval Flow, not the swimlane.

Decision Tables

A rule can be defined using one or more if-then constructs or a decision table. In the Sales Quote example, we use a decision table, and here's what it looks like in the application.
The red highlighted rows refer to whether Business Practices Review is required. Everything else determines the approval management flow for the Approve Deal activity.
This decision table is a compact way to describe a complex set of conditions and actions and may seem a bit overwhelming at first. Take a closer look if you want to, and click Next when you're ready to move on.

Business Rule Global Variables

The decision table you just saw relies upon four global variables, so as part of the implementation of the business rule, Developers need to configure these in the Globals tab of the rule.
If Developers create variables instead of hard-coding values in the rule, then business users can edit these variables at run time without having to touch the rule logic. For example, business users could change the pre-approved discount, or make changes to the tier2Approvers base and high lists.

Business Rule Bucketsets

The decision table also makes use of bucketsets, which define lists or ranges of possible values. You can use bucketsets as a tool in specifying rule conditions.
For example, the Approve Deal rule has a bucketset for quote discounts as shown here:

- >= 90%
- >= 60% and < 90%, and so on.

Approve Deal uses this bucketset in the decision table you saw earlier, and specifies actions for each range.
Here's detail for the RevenueBucket bucketset. The highest range is greater than or equal to $1,500,000 and it goes down from there.
The benefit is, business users can edit bucketsets at run time, thereby changing how rules work without requiring help from the Developer.

Business Rules: Data Input and Output

So once you've defined a business rule and bound it to the activity, you need to map data into and out of the business rule activity. The input for this particular task is the quote object, which contains all the information in the quote. Based on the defined business rule, the output (on the right) contains information related to whether Business Practices Review is required and also various rules on what approvals are required for the Approve Deal task.
Together with the decision function, perhaps you can see how the business rule task takes the quote as input, applies the decision function, and sets the values in the approvalFlow data object.

Course Road Map

So you've seen how business rule tasks are implemented. Now let's look at what's involved in implementing *user* tasks.

What Are User Tasks (Interactive Activities)?

What do we mean by user tasks? We're referring to tasks that require user interaction, represented in the model by green boxes that have a little person icon in the upper left. These are also known as interactive tasks.
For example, the Sales Rep enters a quote, if necessary someone with the BusinessPractices role performs Business Practices Review, various people approve the quote if necessary, someone with the Contracts role approves the terms and finalizes the contracts, and so on.
At run time, when the process reaches each of these interactive tasks, control goes to the Human Workflow engine, which executes the human task and then returns control to the BPMN engine.

Implementing User Tasks (Interactive Activities)

To implement a user task, you need to define a service component called a human task, associate it to the activity, and then map data into and out of the activity. As we mentioned earlier, you should ideally have defined all the necessary data objects beforehand.

Out-of-the-Box Human Task Patterns

BPM Studio offers a number of human task patterns out of the box, to make it easy to implement the most common types of human tasks. For example, the Simple pattern just assigns individual users or groups. The Management pattern is used for a sequential list of approvers up the management chain. Initiator is for the person who kicks off the process, in our case the Sales Representative.

For most of these, swimlane roles are used to determine assignments. For the Complex task, however, the swimlane is irrelevant, and routing and assignments are typically more complicated.

Demo: Implementing a Simple User Task

Let’s start with a simple example: the Approve Terms user task. Take a look at how to implement the Approve Terms task using the BPM Design Editor.

BPM Design Editor

As you saw in the demo, you can use the BPM Design Editor to implement user tasks. Directly from the Properties Implementation tab, you can create a new human task, select from among the most common human task patterns, add parameters to the human task, and map data into and out of the user activity. The resulting human task is automatically bound to the user task.
For many user tasks, this is all you need. But if you want to use more granular human workflow or approval management, or define deadlines, escalation policies, and many more options, you need the *human task editor*.

Human Task Editor: Introduction

Here’s the interface for the human task editor, with the General tab selected. Each tab on the left displays a different view and allows you to edit different aspects of the human task. Click the buttons to learn more, and remember that you can click an image to zoom in and out.

Human Task Editor: General

The General tab is where you define task details such as the title, outcomes, owner, priority, and other attributes. For example, in the Approve Deal human task, the outcomes are APPROVE or REJECT and the task owner is jstein.

Human Task Editor: Data

The Data tab allows you to define the structure of the data in the task, and whether it is editable. In the Approve Deal example shown here, there are three data elements, and one of them (QuoteRequest) is editable. The other two are used to define the approval flow that is needed for a particular quote, based on rules evaluated earlier in the process flow.
The Mapped Attributes section at the bottom allows you to make certain parts of the data available to user worklists for searches and custom views. In the Sales Quote example, if you were to add revenue to the Mapped Attributes section, users could search quotes in their worklist based on revenue and even create a custom column called revenue.

Human Task Editor: Assignment

In the Assignment tab, you assign participants to the task and decide how the task will be routed. For example, here are the assignments for the Approve Deal human task in the RequestQuote process.
There are three stages of participants, called tier3, tier2, and tier1, and they are sequential. Earlier in the process, the Determine Approvals business rule evaluates each quote and determines which tiers of approval are required, if any.

- If tier3 approval is required (the top stage shown here), then the quote is sent for approval *in sequence* to a series of managers, starting with the manager of the Sales Rep who

submitted the quote and going up several levels of management according to the results of the Determine Approval business rule. Notice the icon for the SalesManagement stage shows a serial participant type.
- If tier2 approval is required, the quote is sent *simultaneously* to a list of participants built using "Names and expressions," specifically a group called Tier2Approvers which is defined in the Determine Approvals business rule. Notice the icon for the Tier2Approvers stage shows a parallel participant type.
- If tier1 is required, the quote is sent to the members of the approval group called Tier1ApprovalGroup. This approval group is configured using Business Process Workspace. Notice the icon for the Tier1ApprovalGroup stage shows a participant type of Single, in this case, a single group.

So you see there is tremendous flexibility in defining assignments and routing.

Human Task Editor: Presentation

The Presentation tab allows you to specify resource bundles to display task details in different languages in the Oracle BPM Worklist. It also allows you to select WordML and custom style sheets for attachments.

Human Task Editor: Deadlines

As its name suggests, the Deadlines tab can be used to specify due dates, expiration, and escalation policies. In the example shown, the task is set to escalate after three days to a maximum of two management levels, not to exceed the Director level.

Human Task Editor: Notification

You use the Notification tab to create and send notifications to various types of users. For example, the first row specifies that users assigned to a task should be notified upon assignment. The third row specifies that the task owner (which is specified in the General tab) should be notified whenever there is an error. You could set up notifications when a task completes, or expires, or is withdrawn, and so on. In the Notification Header column, you define just what the notification message should include.
In Business Process Workspace, users specify their notification preferences: do they want to receive notification through email, voice message, instant message, or SMS.

Human Task Editor: Access

The Access tab allows you to specify what parts of a task participants can view and update.

Human Task Editor: Events

In the Events tab, you can specify executable code known as callbacks to execute when a particular stage is reached during the lifecycle of a task, whether the change in state is due to standard routing, reassignment, escalation, or any other reason.

Mapping Data Input and Output

Whether you have used the BPM Design Editor or Human Task Editor to define the human task, you need to check the data associations and make sure they are correct. The application knows what data object types are in the human task and will list them for you in the center area, but you may need to specify exactly which data objects should be mapped to each type as input and output.

Creating and Customizing Forms

Lest we forget, you need to create forms for human tasks, so end users have something to view and possibly fill in. In the Human Task Editor, you can create default forms, or you can customize or add additional forms.

Course Road Map

Let’s talk briefly about implementing service tasks.

What Are Service Tasks?

You use Service tasks to invoke synchronous operations in services and BPMN processes. By synchronous, we mean that when the BPMN Service Engine runs a service task, it invokes the operation specified in the service task and waits for a response. In our example, the SaveQuote activity is a Service task. It represents an automated (or system) invocation step.

Types of Service Components and Service Adapters

You use the Composite Editor to create services, and the SOA resource palette lists the types of service components and adapters you can configure. As you can see, there are lots of options! For example, you could define a BPEL process service if you want to call a BPEL process, a Database Adapter service if you want to write to a database, or a File Adapter service if you want to write to a file on disk.
The Web Service adapter listed at the bottom is generic—it is for web services that are not already listed. With the wizard’s help, you can configure a Web Service adapter from scratch.

Implementing Service Tasks

How do you implement a Service task? Well, you need to define a service and bind it to the activity. The configuration of that service varies according to which kind of service is selected. Then, as always, you need to map data input and output.
As an aside, services are not reserved for Service tasks. They can also be used to implement Message events as well as Send and Receive tasks.

Course Road Map

So you’ve had a glimpse of what’s involved in implementing three types of activities. What else is included in implementation?

Implementation: Other Tasks

By way of review, you need to implement the gateways, create approval groups, map roles, and make other enhancements. Let’s look at some highlights.

Implementing Exclusive Gateways

You may be wondering about how gateways are implemented. In the interest of time, let’s look just at exclusive gateways. Basically, you need to define the conditional transition or transitions, and if there are multiple conditional transitions, you need to specify the order in which these conditions should be evaluated.
Let’s look at an example: the Approvals Outcome gateway and its outgoing conditional transition. The expression in the conditional transition simply states that if either the Approve Quote or Approve Terms outcome equals REJECT, then the process flow goes back to the Enter Quote task. As the gateway has only a single outgoing conditional transition, there is no need to specify an order.

Enhancing the Process Model: Introduction

In the course of implementing a business process, Developers may need to make changes to the process model they get from Process Analysts. This should be done in collaboration with the Process Analysts, of course, to ensure the requirements are being met. In Oracle BPM, it is easy to collaborate, since Process Analysts and Developers share the same process model.
Click Next to learn about various enhancements to the process model.

Enhancing the Process Model: Initializing the Quote

In order to prepopulate the form the Sales Representative fills in, the Developer would add a Script task. Here it’s called Initialize Quote.

Enhancing the Process Model: Setting Business Indicators

For the RequestQuote process, let’s assume there are four company-specific business indicators that the Process Analyst has requested be measured. As with any other data objects, you need to explicitly assign values to these. So the Developer might add a Script task after the Enter Quote task to assign values to three of the business indicators (discount, industry, and revenueDimension) from the quote data object that comes out of the Enter Quote task. The numQuoteEdits counter is set by the Enter Quote task itself, counting the number of times the Enter Quote task is performed, either for creating or revising quotes.

Enhancing the Process Model: Adding Measurement Marks

Once the business indicators have been set based on the quote, you may want to capture these values for the sake of monitoring. To do this, you need to include a Measurement Mark. Wherever there are Measurement Marks, the software collects data on the specified business indicators and on the built-in metrics as well.

Enhancing the Process Model: Adding a Self-Approval Flow

You could add a gateway just before the Approve Deal task to catch quotes that qualify as self-approved and therefore do not need to go through Approve Deal. You could add a Script task here too, to set the variables that will be evaluated further down the line to determine if the quote moves forward or is sent back to the Sales Rep to be revised.

Mapping LDAP Users to BPM Studio Roles

Recall who performs each user task: in most cases, it’s participants who have the role on the left of the process model—the swimlane role. At some point in the process, you need to map LDAP users to these roles.
You can map users to roles *before* deployment using BPM Studio’s Organization Manager. Otherwise, you can map users or change mapping *after* deployment using Business Process Workspace.

Creating Necessary Approval Management Components

The Approve Deal task is a special case, as you may recall. It’s what’s known as a Complex task, and the human workflow is defined in the Approve Deal human task.
Recall that there are three tiers of approval, and that each of them demonstrates a different kind of approval chain. For tier2, you need to define the appropriate global variables. For tier1, you need to create an approval group in Business Process Workspace.

Other Implementation Tasks

Lastly, you’ll need to implement Start, End, and intermediate events, and of course configure error handling and any necessary messaging, including inter-process communication.

Course Road Map

Once you’ve completed the implementation tasks, you’ll want to deploy and test the BPM project before you roll it out into production.

Deploying the Project

You can deploy a project as a SOA composite application from BPM Studio, Business Process Composer, or Enterprise Manager. You need to deploy the project to an application server that is running Oracle SOA Suite.
You’ll want to deploy to a test environment first, of course, so you can ensure the process is working as expected.

Testing the Process: Business Process Workspace

In the test environment, run the process as various sample end users, initiating the process, advancing it through all the possible paths, and making sure the flow follows the expected path and generates the expected set of messages and events. This slide shows a sample screen from Business Process Workspace, one of the most common ways end users participate in an Oracle BPM process.

Debugging the Process: Setting Development Mode (EM)

In testing the process, if things are not working as you expected--process instances take the wrong branch, or the approval flow is not what you were hoping for—you can debug the process by setting the Audit Level to Development in Enterprise Manager, the main administrative tool for Oracle SOA Suite. You may also want to select the Capture Composite Instance State check box so can see the state of each instance—whether it is running, completed, faulted, and so on.
Once the Audit Level is set to Development, initiate new process instances in Business Process Workspace. You’ll see the benefits of Development mode on the next slide.

Debugging the Process: Viewing Audit Trail and Data Values (EM)

Enterprise Manager provides a detailed audit trail of each process instance, as you see here. The blue links are clickable and display data values at that particular point in the process. For example, you could display data values when the instance left the Initiate Quote script task, before the Sales Representative created and customized the quote. Here’s an example.
Or you could display the measurement values that were captured here, after the Set Business Indicators script task. Here’s an example. Maybe these values will help you understand why process instances aren’t branching properly later in the process.
Development mode makes available far more of these data value links, which is why it is so useful for debugging. When you have finished debugging your process and are ready to go into production, don’t forget to set Audit Trail to Production mode, to minimize the impact on performance.

End

Hopefully you understand at a high level what’s involved in implementing a business process in Oracle BPM 11g.
Thanks for participating in this self-paced training, and see you in one of the other online courses in this Oracle BPM 11g series.