Process Implementing with
IBM Business Process Manager Standard V8.5 – I
ZB808
Self-Paced Virtual Classroom (SPVC)

Script

ZB808 Course Introduction

Screen Name: *Course title*
Screen Number: *2*
Script:

Welcome to the IBM WebSphere Education ***Process Implementing with IBM Business Process Manager Standard version 8.5 – Level I*** course.

The host for this course is Ed Barta, a WebSphere Education Curriculum Developer who is based in the US. Ed is here to guide you through the modeling and implementing concepts with IBM Business Process Manager and complete some demonstrations. In this course, you learn about the Business Process Manager tool as applied to modeling a process in a series of instructor walk-through demonstrations.

Screen Name: *Course overview*
Screen Number: *4*
Script:

This course teaches Business Process Management (BPM), methodology, and implementing processes with IBM Business Process Manager, a comprehensive BPM platform that provides the visibility and insight businesses need to effectively control their processes. In this course, you learn about the core process modeling skills. You also learn a project development approach, process model implementation fundamentals, and exceptional delivery patterns. These skills improve the speed and quality of process definition and implementation efforts. Concepts are presented in a systematic way, building upon each other to facilitate a good understanding of process modeling. The Express, Standard, and Advanced IBM Business Process Manager configurations all contain a business processing modeling component that is called the Process Designer. This course focuses on process modeling and implementing with the Process Designer.

Screen Name: *Course objectives*
Screen Number: *5*
Script:

After completing this course, you should be able to:

- Describe why process modeling is an important phase in the BPM lifecycle
- Identify how to use Process Designer to create a process application

- List and identify the core elements that are used to create a BPD in Process Designer
- Translate workflow steps into business process activities and nested processes
- Use gateways to control the process flow
- Validate that the process model meets playback 0 goals and requirements
- Identify how intermediate events are used during the execution of a business process
- Describe the architecture of IBM Business Process Manager
- Organize process assets into toolkits
- Manage variables and data flow
- Implement timer events
- Implement gateways and routing to control process flow
- Build a business data model
- Build services and user input forms (coaches)
- Create a snapshot for deployment
- Create a decision service
- Model and implement message events
- Apply asset tags to organize artifacts
- Enhance coaches for a rich user experience
- Implement effective error handling in processes and services

---

.

Unit 1: Introduction to business process management (BPM)

Screen Name: *Unit introduction*
Screen Number: *1*
Script:

Welcome to the first unit, Introduction to business process management (BPM). Organizations seeking to improve their business processes turn to business process management to provide a systematic approach to accomplish this improvement. BPM initiatives gain momentum from designation of first critical or first launch projects to adoption. To begin this journey towards BPM maturity, the first project must have the right methodology and strategy to set good practices for future BPM initiatives. This unit covers the methodology and strategy that undergirds the development of the cornerstone of BPM: the process model. Regardless of the project, large or small, it all begins with a process model. A few items are required to meet the executive team mandate: a clear model that communicates what the process is and where improvements can be achieved, and a clear model that achieves process efficiency. In the end, a good process model aligns the interest of the business and the execution of the IT groups. Process models that connect to all the phases of project development and deployment allow for key gains in change management agility.

This unit covers the details on process models, how models are developed and used, and what development strategy is employed. During the course, you learn how the business process management tool delivers these models in the best possible manner. It all begins with a discussion on business process management and the business process management lifecycle methodology to level-set the foundation for process modeling efforts.

Screen Name: *Unit objectives*
Screen Number: *2*
Script:
This unit reviews the foundational concepts that establish the importance of process modeling. This unit includes a review of business process management, the business process management lifecycle, the basics of process modeling, and business process management project development. The unit introduces a case-study scenario that begins the in-class development of a business process definition that is based on the business requirements that are established within the process analysis - playback zero phase of the project.

After completing this unit, you should be able to:

- Define business process management (BPM)
- List and describe the phases in the BPM lifecycle methodology
- Define process modeling

- Describe playback zero and the achievements that are reached during this stage of project development

---

Screen Name: *Unit 1: Introduction to business process management*
Screen Number: *3*
Script:
The topics for this unit are:

- Business process management (BPM)
- About process modeling
- BPM project development
- About playbacks, specifically playback 0
- The Hiring Requisition process

You begin with the topic of business process management (BPM).
Companies often seek ways to improve their organization to increase productivity, lower cost, and increase revenues. The challenge organizations face is that change in the way business is conducted is inevitable because of various factors, like market dynamics. To keep up with the climate of change in business, organizations rely on efficient and effective business processes.

Many companies try to implement different strategies to accomplish change management of processes with little to no disruption of customer service and employee productivity. But many times, those efforts either fail outright or accomplish only a portion of the wanted process improvement because there is little performance measurement.

Screen Name: *What is BPM?*
Screen Number: *4*
Script:

Business process management (BPM) excels at providing a comprehensive change management of business processes that results in continuous process improvement.

Screen Name: *Three themes*
Screen Number: *5*
Script:

BPM is also described by using these common themes: the goal, the system, and the expected results.

- The BPM goal is efficient business processes with visibility.
- The BPM system is the management of people-to-people work steps, system-to-system communications, or person-to-system interactions.
- The BPM expected result is process improvement that brings about financial benefits and customer and employee satisfaction.

Screen Name: *BPM vision*
Screen Number: *6*
Script:

Does BPM have or even require a vision? Is the ideal to match existing core competencies and toolsets with the stated goal, system, and expected result? If so, then vision might be limited to only the execution of a process application build. This vision would also affect what a process model might look like and what it would communicate to an application development team.

What if a broader vision for BPM is the following?

“BPM is the means by which companies and governments improve their operations by using internal business expertise in new, scalable ways. This objective is achieved by directly engaging business people in the design, definition, and creation of enterprise-class process applications.”

This vision provides a wider scope for BPM. It specifies the need to change not only the business process but also the strategies, methodology, and most importantly, the thinking in project development.

Screen Name: *BPM lifecycle*
Screen Number: *7*
Script:

An illustration of the BPM lifecycle is provided. Its four phases are design, modeling, execution, and optimization.

Looking at the BPM lifecycle, one thing becomes apparent. There are opportunities to use the expertise of business and IT to collaborate in each phase of the lifecycle. Using this approach to BPM, the business process is stable and on-target. This stability is because of the overall iterative improvement cycles. The cycles keep up with business goals, business change, and opportunities within each phase to make critical adjustments. Business and IT working together throughout the BPM lifecycle requires a clear set of goals for each

phase. Matched against those lifecycle phase goals are the responsibilities for each group. Clearly, the governance of the business process varies at each phase for each group. But, the involvement of both business and IT ensures that the process improvement is realized.

Screen Name: *The modeling phase*
Screen Number: *8*
Script:

This portion of the course focuses on the modeling phase of the BPM lifecycle and how business and IT collaborate to create the process model. The modeling phase is more than just creation of the process model. This phase also requires an understanding of how to adjust the model to meet evolving business requirements. So throughout this phase, the process model goes through continued analysis and a series of adjustments and refactoring efforts. The result is a model that can be implemented into a process application.

All the adjustments and testing allow for a process model that meets what the business expects. Deliver an improved and efficient business process at project development end. BPM done correctly results in business processes that are modeled, analyzed, and adjusted early and often. The BPM effort goes far beyond applying technology to a process to yield a changed process. Applying technology to automate a bad business process without regard to appropriate analysis and adjustment efforts leads only to an efficient, but still bad process.

Screen Name: *Unit 1: Introduction to business process management*
Screen Number: *9*
Script:

Understanding a business process is the next topic.

Screen Name: *What is a business process?*
Screen Number: *10*
Script:

A business process is a set of tasks or activities that takes specific inputs and converts them into specific outputs in a wanted, predictable fashion. Inputs are typically information or a set of information that triggers a set of activities in the process. Outputs are the decisions that render these activities.

Screen Name: *What is process modeling?*
Screen Number: *11*
Script:

Process modeling captures the ordered sequence of the business process tasks or activities. Process modeling also captures the responsible roles that are doing the activities. It captures conditional branching, the sequencing of the flow of work between activities, and all supporting information from start to end.

Screen Name: *Three-phase approach*
Screen Number: *12*
Script:

Process modeling can be described as having a three-phase approach:

- Phase 1 or descriptive: Describe the process. The process is a high-level model that describes the process that is based on business requirements. The model is easily communicated across the organization.
- Phase 2 or analytical: Analyze and improve the process. The process is analytical, more detailed modeling, showing all pertinent activities and flows that are used to detail process requirements.
- Phase 3 or executable: Implement the process. The process is a model that details the functional requirements and implements the executable process application.

The first part of this course covers the first two phases: descriptive and analytical.

Screen Name: *What is a process model?*
Screen Number: *13*
Script:

A process model is a graphical representation or diagram of the business process. A good process model is easy for business people to use and understand, and it is directly implemented in a business process management system (BPMS). Process owners and process participants (business) must recognize the same concepts in the same context. BPM development teams must recognize these concepts as well for all parties to universally understand a process model. There is no need for IT to redraw a process model to provide more clarity or a different point of view.

A good process model provides views into a process that are clearly and easily communicated in 5 minutes or less at every level of granularity. After you

establish when and what must be accomplished in process modeling, the next focus is how process modeling is accomplished in terms of methodology. Understanding how to model a process requires comprehension of the BPM project development methodology. Realize that project development strategies for process models differ from standard methods, especially when you consider the usage of process models. In standard project development, the longevity of diagrams that is derived from requirements exists only from the business hand-off to the development teams. In essence, these requirements are translated into code and their use ends at that point.

BPM process models are different. Examine the BPM lifecycle, and notice that the process model evolves in terms of usage. The prolonged longevity of a process model allows for the iterative BPM lifecycle because it is data-driven and not code-driven. This model provides the stability for a process application without fear of having to start from scratch when change is needed. This data-driven process model is known as a business process definition (BPD) in Process Designer.

Screen Name: *The right process modeling development strategy*
Screen Number: *14*
Script:

Business Process Manager uses a single shared environment for project design and development. All process artifacts are stored in a single shared model architecture. All parties that are involved in the effort to define, model, implement, measure, and improve the process are working from a common shared platform. That platform encapsulates all of the various components. This shared platform keeps the vision of bringing business into the same room as IT intact.

The business people who use the application and the developers who create the application are all using the same business process definition, or process model. Analysts and developers build the model of the process, and that same model is the model that completes at run time. The same model is used to create reports on the performance and status of the process, and the same model is used to implement process improvements.

Screen Name: *Topic 3: BPM project development*
Screen Number: *15*
Script:

The next topic is BPM project development. Project development for any IT initiative typically reinforces established standards and methodologies. A BPM project, especially one that includes the broader vision and definitions that are provided, would not fit the typical project development standards. It is because the critical BPM project components are slightly different. The BPM project

components reinforce process first, solution second. In BPM, everything is process-driven; the methodology that is used to develop process models and the process application are uniquely focused on the process needs. The IT development team traditionally develops projects by using a system that fails to focus on process first.

The next topic covers the BPM project development strategy that fits the business process management needs.

Screen Name: *BPM project components*
Screen Number: *16*
Script:

The top-down diagram view of the BPM components provides a quick view of how a typical BPM project development evolves. Any of these components that are missing from a project would interrupt the effective design, definition, and creation of the process application. It would also curtail the engagement of business people.

What are these components?

- A project vision
- Project goal
- KPI and metrics
- Business requirements
- Process requirements
- Functional requirements
- Development requirements
- Solution implementation

Knowing how to navigate through each component of a BPM project helps to ensure a successful process modeling effort.

Screen Name: *Project development phases*
Screen Number: *17*
Script:

In contrast to the traditional IT application development approach stands the phased BPM project development approach that aligns to the overall BPM lifecycle methodology. This approach also focuses on the BPM project components and allows for the different phases of process modeling.

- Definition: Discover and define the process. Analyze for improvements. Model and set the process performance measurement criteria
- Development: prepare the process application for deployment by using an iterative development playback system
- Test: validate that the process application performance achieves expected business process goals in a production platform user environment

---

Screen Name: *BPM project teams*
Screen Number: *18*
Script:

Before you talk more about the process modeling effort, it is a good idea to talk about BPM project teams. Integral to the components within a BPM project are the roles and responsibilities for each. For example, for a clear articulation of the BPM project vision, you seek out the process owner, sometimes known as the BPM sponsor or champion. It is important to know the different BPM roles for process modeling because collaboration is critical for all project team members. A key feature of IBM Business Process Manager is the built-in function to facilitate collaboration throughout the project development lifecycle.

The unique phases and components of a BPM project require a specific set of project roles, including:

- Process sponsor: responsible for establishing the project goals and scope, securing organizational support and resources, and ensuring alignment with organizational business goals
- Process owner: person who knows everything about the process
- BPM project or program manager: person responsible for the project success
- Subject matter experts: Person with knowledge of specific process resources, or systems
- Core BPM project development team that typically includes:
    - A BPM analyst

  - BPM developer (includes integration designer developers and technical consultants)
  - Solution architect (advanced role that can lead teams and serve as an analyst and developer)
- Administrator: Install, update, and configure the business process management system
- Facilitator: (optional) person who typically manages the collaboration meetings for a BPM team

---

Screen Name: *Unit 1: Introduction to business process management*
Screen Number: *19*
Script:

The next topic covers specific information about playbacks. Specifically:

- What they are
- How they are managed
- What benefits are gained by using playbacks as part of the project development strategy

---

Screen Name: *Playback iterations*
Screen Number: *20*
Script:

A playback between business and IT is a focused demonstration of a partially implemented process model at the designated development phase. The goal of a playback is project development discussion, consensus building, collaborative improvement, and ultimate approval of the process model. Playbacks also enable the iterative development of the process application. Here is an illustration of a playback iteration.

The use of playbacks iteratively provides early visibility and input from the business group on the process application functions. The feedback from the business group during early stages of a project benefits development. The business group identifies adjustments to requirements well before the final product is implemented. The ability to shift direction during the definition and development phase is necessary for reaching the ultimate wanted BPM project target.

Iteration of the process model is a critical factor for BPM success.

Screen Name: *Playback cycles (themes)*
Screen Number: *21*
Script:

Often playbacks are conducted as themed stages. The number of actual collaboration meetings that are conducted during each stage can vary. Usually, multiple smaller playbacks are conducted and target individual groups with specialized roles (for example: analyst, developers, or administrators). But these smaller playbacks build upon one another leading to a final playback. During this final playback, project teams reach consensus before you move to the next stage.

This diagram is an example of a project plan that contains typical definition and development playback timelines. In modeling and implementation, shorter cycles are a good practice. The agile development approach with shorter cycles is a BPM success factor. Larger projects are scoped down to smaller release cycles. Creating smaller chunks for a project has many benefits that include:

- Overall project risk is reduced.
- Working code can be released into production in a shorter amount of time.
- Changes to the project cause less rework if releases occur on a more regular basis.

Screen Name: *Playback 0*
Screen Number: *22*
Script:

The definition stage of project development focuses primarily on descriptive and analytical process modeling. Because of this focus, there is a playback zero for the definition stage. The goal for playback zero is to iterate upon concerns about business process effectiveness until a good working solution is reached. Then, the process can move on to implementation.

Before a process is modeled, it must be fully discovered. That requires sessions with the business process owner to uncover the particulars of the business. As the process is defined, it is necessary to analyze and create initial models by using an incremental approach. From the current state, the model changes to the wanted business process that is validated as a final “To Be” process model.

Screen Name: *Playback zero: Process discovery (1 of 2)*
Screen Number: *23*
Script:

Early stage, or descriptive, modeling is based on discovery and analysis. It has an outcome that both the process owner (business) and BPM team designate as the current state business process. The process discovery effort in playback zero captures the initial process information that translates into the initial process model. Process owners and BPM team members undertake the effort of process capture. They all want to make sure that the current state of the business process is documented. This documentation can be stored in various tools available to the team, including Blueworks Live. Blueworks Live has an advantage over other tools by providing connectivity to IBM Business Process Manager. Additionally, with Blueworks Live, the documentation effort maintains a high level of usage, even beyond process discovery and analysis of the business process.

Screen Name: *Playback zero: Process discovery (2 of 2)*
Screen Number: *24*
Script:

Moving from discovery to model
Business process discovery documentation, or mapping, with as much process detail as possible is a quick process, and the beginning of process modeling can happen early. So, the most common question when in the midst of a process discovery effort is:

When is the move from process discovery to process modeling?

Several aspects need to be considered to answer this question. If the process discovery sessions exhaust all requirements to communicate everything about the process, then it is time to transfer over to a process model. Also, consider the conversations in the process discovery sessions during playback zero meetings.

- The question no longer is, “What does this process do?”
- The question is, “What does this process look like?”

The move to a process model is near. This move is when an “As Is” model can be developed for the business process.

Screen Name: *Playback zero: Process analysis*
Screen Number: *25*
Script:

The next phase in process modeling and playback zero is analytical modeling. The milestones to get to the final stage in analytical modeling are analysis and the “To Be” model.

Discovery documentation is a “just the facts” effort. The only thing that interests the BPM analyst and process owner is that the process is captured in its current state. A good example is the capture of process problems, or pain points. During discovery, it is not important to try to solve process pain points, only to document them. Now is when the next step in playback zero starts to take shape: process analysis.

Process analysis occurs when the business process is continually refined until analysis goals are reached, such as solving process pain points through root cause analysis. Other process analysis goals are the added value analysis of each process activity that is captured and analyzed to ensure the appropriate priority for improvement opportunities. With process analysis, the business requirements are vetted and the process requirements are aligned so the “To Be” process model is ready to be finalized.

Screen Name: *Playback zero: Iteration*
Screen Number: *26*
Script:

Playback zero has a unique set of achievements during this stage of project development. A BPM analyst works with the process owner and BPM project manager to handle most of the goals in playback zero. BPM project managers need the data from playback zero to plan for the next project development cycles, or playback stages. The process owner is interested in the business process. For a process owner, the outcome of playback zero must have the most efficient and effective business process model for continued use in implementation.

To achieve both objectives, the strategy that is used to gather playback zero data is the iterative phase approach of:

- Capture: Make sure that the business process information is shared fully.
- Map: Create a Discovery Map that can clearly define the important information in an easy to read manner.

- Document: Refine documentation as the analysis continues as there are more stakeholders than just the process owner.
- Refine: Allow for adjustment to a business process as a clear definition of the business process and process model is incrementally made.

Screen Name: *Business data*
Screen Number: *27*
Script:

Analytical modeling allows the process owner and BPM team to identify the business data that is needed for the process model. In general, business data provides the context of the process task for each process participant role for process task completion. For example, the business process is to process an insurance claim. The task is to verify the claim. The business data provides the claim type, claim number, claim description, and claim submitter.

When you define the business data for a process model, BPM teams look at the process as a whole. The question for each activity becomes: “What data does the process need to complete this task?” In the end, the process analysis produces the “To Be” process model and a business data object model as well.

Screen Name: Unit 1: Introduction to business process management
Screen Number: *28*
Script:

The final topic in this unit is the hiring requisition process. The Hiring Requisition process is included in the Process Modeling and Implementation course student notebook. This process narrative is provided to enhance your learning experience. Challenge activities are also provided in the student book for this course, and they provide more learning objectives to the core requirements listed. The challenge requirements are not required to complete this course. They are optional. They are intended to help you extend your knowledge. The challenge requirements take you beyond the content in the book and the exercises you complete as part of the course.
This product is the result of an extensive process of gathering requirements and went through process decomposition. Read through the Hiring Requisition process on page 1-48 in the book and 1-2 of the exercise guide, and become familiar with the scenario you model during the exercises later in this course.

Screen Name: *Unit summary*
Screen Number: *29*
Script:

Having completed this unit, you should be able to:

- Define business process management (BPM)
- List and describe the phases in the BPM lifecycle methodology
- Define process modeling
- Describe playback zero and the achievements that are reached during this stage of project development

Unit 2: Introduction to IBM Business Process Manager and Blueworks Live

Screen Name: *Unit introduction*
Screen Number: *1*
Script:
Welcome to unit 2: Introduction to IBxM Business Process Manager and Blueworks Live

---

Screen Name: *Unit 2: Introduction to IBM Business Process Manager and Blueworks Live*
Screen Number: *2*
Script:
With a vision in place to start a BPM project, the next task is to find the right toolset. It is not only to build the process application, but to maintain the project lifecycle synergy between the business and IT when it comes to process modeling.

This unit focuses on the best tool: IBM Business Process Manager. This unit is a review of how Business Process Manager fits within the strategy of playbacks. It also includes the product features that enable process modeling, sometimes called authoring, and the peripheral applications that support the modeling effort.

After completing this unit, you should be able to:

- Describe how IBM Business Process Manager is used to accomplish process modeling goals
- Explain how to create and modify process applications in the Process Center
- Explain how to create and modify process models with the Designer view of the IBM Process Designer
- Describe how to validate process models with the Inspector view of the IBM Process Designer
- Describe the purpose of the Process Portal
- Describe the purpose and function of Blueworks Live

Screen Name: *Topic 1: About IBM Business Process Manager*
Screen Number: *3*
Script:

The first topic is about IBM Business Process Manager.

As covered earlier, process modeling is a three-phased approach: descriptive, analytical, and executable. Business process management tools strive to meet those objectives in process modeling. The tools must also fit within the parameters of agile development and collaboration between business and IT. The next topic covers the best tool for accomplishing those objectives: IBM Business Process Manager.

Screen Name: *IBM Business Process Manager v8.5*
Screen Number: *4*
Script:

IBM Business Process Manager is an enterprise application that:

- Allows you to build faster, work more efficiently, and constantly improve your business process model
- Employs a graphical process development tool, Process Designer
- Provides process visibility and performance data
- Provides process simulation and optimization
- Provides an interface console for user productivity

Screen Name: *IBM Business Process Manager configurations*
Screen Number: *5*
Script:
IBM Business Process Manager offers three different configurations (Express, Standard, and Advanced). Each configuration has unique capabilities. For example, for the Express and Standard configuration, the Integration Designer is not included. For Express, there is a limitation in author and process participant access, and a processor cores limitation with no clustering.
This course covers process design and implementation, which is performed on every configuration of IBM Business Process Manager. WebSphere Education also offers courses that focus on the other tools included in the advanced configuration like the Integration Designer and the Enterprise Service Bus (ESB). More information is provided in your book on the other WebSphere courses offered.

Screen Name: *How does IBM Business Process Manager fit in process modeling?*
Screen Number: *6*
Script:

The BPM lifecycle requires a BPM team to designate business process

candidates. It is appropriate to choose the best tools to accomplish discovery and definition. One of the best in market tools for discovery and initial definition of a process is Blueworks Live. Eventually, a BPM team is required to move to a tool that handles the adjustments and refinement of the process model on the way towards execution. Remember, it is best to think of a shared model approach to maintain a central artifact to modify and improve the process. This approach is where IBM Business Process Manager excels. The tools do not have a clear demarcation where one stops and the other begins in terms of the process modeling. That varies from project to project. Blueworks Live and Business Process Manager used together engage the business and information technology (IT) sides of an organization. Business Process Manager offers the ability to efficiently handle critical components of a BPM project and the three phases of Process Modeling. Part of the functional charter of good BPM project development involves a strategy that is called “playback”. Playbacks provide the opportunity for business people to collaborate and be involved in the modeling phases. Playbacks also validate work or adjust requirements in the implementation of the model.

For purposes of this course, the authoring starts with modeling in Business Process Manager in the analytical modeling effort.

Screen Name: *IBM Process Designer*
Screen Number: 7
Script:

Process modeling in IBM Business Process Manager is accomplished through the Process Designer views or interfaces. These interfaces allow developers or authors to create, manage, test, and optimize process models. When you model a process in the Process Designer, project teams are creating a business process definition (BPD). A BPD is the reusable shared model of a process, defining what is common to all runtime instances of the process model.

IBM Business Process Manager Process Designer is composed of three key interfaces:

- Designer: This interface of the process designer allows authors to create, modify, and implement business process models and services.
- Inspector: Project development authors can test process models at any time during modeling and implementation efforts to check and debug each business process definition.
- Optimizer: This interface allows project development teams to evaluate the executable process model. The evaluation happens either in a simulated set of business process scenarios, or through historical data that is gathered after deployment.

To access these interfaces, an author goes through the central repository and the Process Center to open or create a process application.

Screen Name: Unit 2: Introduction to IBM Business Process Manager and Blueworks Live
Screen Number: 8
Script:
Topic 2: The Process Center. The unique design environment of Business Process Manager includes a central repository that is called the Process Center. The next topic covers the Process Center essentials.

Screen Name: The *Process Center: The center of process development*
Screen Number: *9*
Script:

The Process Center provides project development teams a shared development platform and an end-to-end governance of process applications to mitigate typical application development problems. The Process Center allows for the management of the different process applications that are created for specific projects along with the assets for each. The Process Center includes a Process Center Server and a Performance Data Warehouse. You can use these components to build and run process applications and also store process performance data for testing and playback purposes during development efforts. Multiple authors can connect to the Process Center and concurrently edit the same processes or assets.

Screen Name: The *Process Center: Process applications*
Screen Number: *10*
Script:

In Business Process Manager, process applications are the containers for the process models (BPDs) and supporting implementations. BPM analysts and developers create the process applications in the Process Designer. Supporting the implementation artifacts are services and toolkits. Process applications can be either newly created in the Process Center or exported and imported.

Screen Name: *Unit 2: Introduction to IBM Business Process Manager and Blueworks Live*
Screen Number: *11*
Script:

The next topic is the IBM Process Designer

Screen Name: *Demonstration – Process Designer*
Screen Number: *12*
Script:
The following is an instructor walk-through demonstration of the process designer. Click the start demo button to begin.

S***creen Name:*** *Exercise 1*
Screen Number: *13*
Script:

Exercise 1: Creating a process application

Screen Name: Unit 2: Introduction to IBM Business Process Manager and Blueworks Live
Screen Number: *14*
Script:
Topic 4: The Process Portal. The Process Portal is the main tool that process participants or business users interact with to complete tasks and processes. Other tools, such as the IBM Business Process Manager mobile application can also be used to complete tasks and processes.
The Process Portal also has use for project development, especially in terms of validation. When BPM teams and business stakeholders want to reach consensus in the playback session to end a stage of development, the Process Portal tool is used. The Process Portal allows the team to view the process performance as it would function in a user environment.
This unit focuses on the Process Portal, but also shows the mobile application to display other ways business users can run processes and tasks.

Screen Name: *The Process Portal*
Screen Number: *15*
Script:
The main page to run and manage tasks is the Process Portal. You can view your performance or your team's performance by selecting a report in the Dashboards menu.
Overdue and Due Today headings separate the tasks. You can manage tasks by using the menu directly to the right of the task. To complete a task, click the step and claim the task. You can complete the task on the next page.

You can search tasks with the search feature, and below that you can also filter to look at tasks that were created in the past.
On the right side of the window is where you can create an instance of a new process.

Screen Name: *Process Portal: Social features*
Screen Number: *16*
Script:
The Process Portal has several social features. When you complete a task, access task details are presented in a menu in the right window. The process activity stream, and experts of the task are also shown.
You can use this menu to comment and post in addition to read and comment on what actions are the process completes immediately after they occur. You can upload a photo to show next to each post. Automatic system posts are designated by an icon with blue gears.
In addition to text posts, you can also post attachments and links. Mention other users by using the @ symbol and typing the first few characters of the user name.
A developer can assign experts, or the system identifies after a process completes several times.

Screen Name: *Process Portal: Mobile application*
Screen Number: *17*
Script:
Depending on the company policies, users might not use the Process Portal. Email, Business Space widgets, or integrated software of a company might replace the Process Portal for your organization. Mobile applications dominate the way businesses are run. Apple devices have a native application that can be downloaded from the iTunes store to complete user tasks with a custom portal that is built for their device.

Coaches are displayed in the mobile format by using custom mobile-ready coach views. These mobile coaches can be built for optimized display on iOS or Android devices.

Screen Name: *Process Portal: Worklight integration*
Screen Number: *18*
Script:
IBM Business Process Manager bundles IBM Worklight to create mobile-ready pages for Android and iOS devices. Worklight takes a hybrid approach to

developing mobile applications that take advantage of IBM Worklight's cross environment developer productivity and server-side features for security, scale, and management. Developers build pages in native HTML, build only one page, and it is cross-compatible for all mobile devices.
Worklight requires a separate license purchase only for production environments. More information on WebSphere education classes in Worklight is included in your book.

Screen Name: *Demonstration – Process Portal*
Screen Number: *19*
Script:
The following is an instructor demonstration on the portal. Click the start demo button to begin.

Screen Name: *Topic 5: About Blueworks Live*
Screen Number: *20*
Script:
As you know, Blueworks Live is an IBM tool that is used primarily for modeling and analysis of processes. The next topic explores the tool and a few of its capabilities.

Screen Name: *How does Blueworks Live fit in process modeling?*
Screen Number: *21*
Script:
Blueworks Live is one of the best in market tools for discovery and initial definition of a process is Blueworks Live.
Companies use a common pattern to model and complete process analysis in Blueworks Live. Then, processes are moved into IBM Business Process Manager for implementation and integration with different systems.
It is important to note that both tools do not have a clear demarcation where one stops and the other begins in terms of process modeling. That varies from project to project; however, both work together to engage business and information technology (IT) sides of a business.

Screen Name: *Blueworks Live and SaaS*
Screen Number: *22*
Script:
Blueworks Live is different than many other IBM products because it is sold as software as a service, or SaaS. Customers pay a monthly fee to subscribe to the service.

Every 8-12 weeks, a new release of Blueworks Live is completed. Many times, the developers rely on suggestions from the customer community to create and add new features. When there is an update, you do nothing, your software is automatically updated.
Blueworks Live requires only an internet connection and web browser to get started modeling. There is no installation of products or maintenance hassles. It is a quick way to start mapping your processes.

Blueworks Live can be used for modeling and analyzing processes, but it can also be used for running small simple processes.

Screen Name: *Collaborative and social modeling*
Screen Number: *23*
Script:
IBM Blueworks Live has many collaborative and social features for modeling and completing processes.
The product has real-time editing, where you can view others changes as they happen. There is also a version history available if you need to revert to a past version or undo changes.
Many social features such as streams, commenting, messaging, and sharing are all available to connect people to get the best process model possible.

Screen Name: *Using IBM Business Process Manager and Blueworks Live together*
Screen Number: *24*
Script:
Many processes are complex, need to integrate with existing systems, and have other functions that are implemented. When analysis is complete, these processes move to IBM Business Process Manager where a team of developers implements the business process model.
In IBM Business Process Manager, you can subscribe to a Blueworks Live account and bring over processes to begin implementation.

Many customers use Blueworks Live for all of their modeling and then follow this sequence. Others use different modeling tools and import those models into IBM Business Process Manager for implementation.
In some cases, small and non-complex process can actually stay in Blueworks Live for their life and be run through email. Many different options exist for modeling and implementing those models to allow various customers to implement Business Process Management.

Note: More courses on Blueworks Live are also available. WB/VB009, WB010, and WB011 offer a process discovery and analysis workshop that includes work in Blueworks Live.

Screen Name: *Unit summary*
Screen Number: *26*
Script:

Having completed this unit, you should be able to:

- Describe how IBM Business Process Manager is used to accomplish process modeling goals
- Explain how to create and modify process applications in the Process Center
- Explain how to create and modify process models with the Designer view of the IBM Process Designer
- Describe how to validate process models with the Inspector view of the IBM Process Designer
- Describe the purpose of the Process Portal
- Describe the purpose and function of Blueworks Live

Unit 3: Create a process model

Screen Name: *Creating a process model*
Screen Number: *1*
Script:

The preliminary process requirements contain the information necessary for stakeholder validation of the high-level process. The validation sessions with stakeholders are called playbacks. During process playbacks, the preliminary requirements are iterated upon throughout the stages of the process application development. In this unit, the objectives are to make sure that the process is modeled or diagrammed in its current state. Another objective is to be precise on the expected order of steps for a stakeholder playback meeting that occurs later.

This unit uses the Business Process Manager Process Designer component for this preliminary model. At this stage, a company might choose to model its processes in another software package. Alternatively, the company might choose to map, model, and import its process diagrams from Blueworks Live into the IBM Process Designer. After creation of the preliminary model and initial playback, the goal is to refine the business process definition. Re-examine the process and identify places to adjust and refine requirements for final implementation.

The core element flow chart-based notation is used to communicate the process to the BPM team and stakeholders. A refined diagram allows BPM teams to create the functional requirements for process application implementation. After this unit, the refined diagram is closer to completion, but recognize that the process develops iteratively throughout its lifecycle and changes as requirements are refined. Continuous process improvement and project development agility are a critical focus of business process management.

Screen Name: *Unit 3: Creating a process model*
Screen Number: *2*
Script:

After completing this unit, you should be able to:

- List and describe the core notation elements that are used in IBM Process Designer
- Examine a defined workflow from detailed process requirements and identify the interrelated process activities and the roles that are responsible for completing them
- Decompose activities into processes and nested processes that contain process tasks

- Create a business process definition (BPD) from the process and nested process tasks and responsible roles

---

Screen Name: *Topic 1: Creating a process model*
Screen Number: *3*
Script:

Creating the process model is part of the effort within playback zero. Naturally, modeling does not stop with creation of the initial process model after discovery. It continues on through iterations until the business process reflects the wanted improvements and adjustments. Continuous improvement is why the shift from descriptive modeling to analytical modeling happens: to produce the process model that is deemed worthy of implementation. The basics of creating a process model from the discovery perspective are covered in the next topic. It is necessary to understand the initial work that is done in the descriptive modeling phase to realize what improvements are applied later.

Screen Name: *Creating the initial process model*
Screen Number: *4*
Script:

The process application implementation begins with the creation of a process model. The process model has an initial focus: capture the current state of the business process and model it to set the foundation for the next phases in modeling.

To understand how to create a process model that is incrementally improved through the playback sessions of the business process, follow these guidelines:

- If you are creating a process model during discovery of the business process, the process model reflects the captured data.
- The process model does not concern itself with solving process pain points (problems) until analytical modeling.
- The process model is agile enough for continued adjustments, so focus first on the expected order of process tasks that is reflected in the model.

Screen Name: *Automation*
Screen Number: *5*
Script:

Careful attention is given to the focus of the initial process model. Many times, organizations erroneously believe that BPM is about adding technology solutions

to process problems as the first order of business. Soon after the initial sessions to document the business process, decisions to automate process tasks can be prematurely made. Automation certainly is a good thing in BPM; however, it is not the focus of modeling at this early stage. Allow the analytical modeling effort to designate opportunities to automate tasks. Opportunities to automate are realized as the iterations on the process model continue.

Screen Name: *Where to start*
Screen Number: *6*
Script:

Where does process modeling start for a BPM team?

Process modeling is a diagram that reflects the ordered sequence of activities within a business process along with supporting information from start to end. In modeling, the business process is framed by using:

- A workflow model to reflect process activities
- The roles that are completing those activities
- Conditional branching
- The sequencing of the flow of work between activities

So a BPM team starts by adhering to the standards used in process modeling, Business Process Model and Notation, which is known as BPMN.

To communicate a process model clearly within an organization, the BPMN notation standard must be applied. A BPM team uses a development methodology that works best to collaborate on modeling with the business groups. This effort would be concurrent to using BPMN to model the business process. This methodology is the playback methodology.

Screen Name: *About BPMN*
Screen Number: *7*
Script:

The primary goal of Business Process Model and Notation (BPMN) is to provide a notation or diagram element that all business users readily understand. Examples of these business users are:

- Business analysts who create the initial discovery maps of the process
- The developers responsible for implementing the process
- The business people who manage and monitor the process

BPMN creates a standardized bridge for the gap between the business process design and process implementation. Multiple BPM vendors agreed on this single notation for the benefit of the user community.

IBM Business Process Manager Process Designer authoring uses six core BPMN 2.0 elements:

- Pool
- Lane
- Event
- Activity
- Flow
- Gateway

IBM implements and interprets these elements to have specific meanings and terminology in the Process Designer application. Later in this unit, core element descriptions explain how Process Designer implements and uses the core elements to implement BPMN.
For broader definitions of the BPMN specification, see the Object Management Group BPMN Specification document version 2.0.

Screen Name: *Process modeling guidelines in IBM Process Designer*
Screen Number: *8*
Script:

These guidelines help to author process models in Process Designer:

- A process model is called a business process definition or diagram (BPD) in Process Designer.
- A subprocess is called a nested process or nested BPD in Process Designer.
- In general, a BPD is as simple an abstraction as you can make because a highly conceptual BPD is resilient to change.
- Authors use the Documentation area in the Properties tab for each BPMN element in Designer to include important requirement notes.

Screen Name: *Demonstration – Creating a BPD*
Screen Number: *9*
Script:

It is now time for a demonstration. To begin the demo, click the Start Demo button.

Screen Name: *Exercise 2 – Creating a business process definition (BPD)*
Screen Number: *10*
Script:
It is now time for a hands-on exercise. To begin the exercise, click the Exercise 2 link immediately following this unit in the Course Outline window from which you launched this unit.

Screen Name: *Topic 2: Pool and lanes*
Screen Number: *11*
Script:
The discovery and analysis session provides details about the business process that can be converted into BPMN process model elements. These elements are used together to create a diagram that describes the business process and later runs the process application. Two specific BPMN elements are covered in the next topics: the pool and lanes.

Screen Name: *Pool*
Screen Number: *12*
Script:

A pool is a graphic element that surrounds each business process model. A process that you model in the IBM Process Designer includes the default IBM Process Designer pool. The pool consists of lanes that you designate beyond what is provided as the default lanes. In essence, the pool is the BPMN element that represents the entire business process definition. This BPMN element is the only one not found in the IBM Process Designer element palette and does not have properties. It is, however, the default setup for all models.

Screen Name: *Lanes*
Screen Number: *13*
Script:

Lanes are horizontal blocks in the element palette, and authors are free to add more lanes to the BPD as needed. There is a concern with too many lanes for a BPD because the purpose of a process model is to communicate the business process effectively.

Each lane represents a group of process participants, the process task responsible role that is captured in the discovery and analysis session. Lanes

provide context for a process model. Each lane contains a series of activities that are assigned to a specific process participant or events that transpire in the process.

To obtain the details for participants during discovery and analysis, BPM teams employ user stories. The user stories help determine which participants are responsible to complete specific process tasks. Each of these process participants is assigned to a lane when you model the process. It is important to remember that a participant is a role, and not a person, in a process model.

Screen Name: *System lane*
Screen Number: *14*
Script:

When it comes time to define process tasks that are automated, there must be a way to communicate this automation in the process model. A participant that is assigned to a lane does not always need to be a responsible human role. Process participants that are assigned to lanes can also be systems.

For example, the discovery and analysis session captures a process task. One task might be completing a background check on a loan that a human does not touch, only the system completes the task. Process Designer has a specific default lane to contain these sorts of automated tasks: the system lane.

During the initial process model build, captured process tasks that are automated are represented as part of the system lane. Further automation of an activity is designed as the process is improved and validated through the iterative playback project development. Process improvement and automation mean system lane rearrangement from one BPD to another to indicate where efficiency is found for the entire business process.

Screen Name: *Unit 3: Creating a process model*
Screen Number: *15*
Script:
A phase is the highest level demarcation phase in a business process. When you start the process discovery, phases are often gathered first because they are the details that process owners easily recall about a process. After phases are established, the process details that fall under each phase are gathered to help complete the discovery and analysis of the business process. The next topic covers the high-level details of phases.

Screen Name: *How are phases used in Process Designer?*
Screen Number: *16*

Script:

When you use the Process Designer, your company might or might not choose to use phases. Phases do not provide functions for process implementation in Process Designer, but are a good organizational framework for descriptive and analytical process modeling. Phases are common in many process discovery and documentation products such as Blueworks Live. If a process diagram is imported into the Process Designer from tools like Blueworks Live, the diagram automatically carries phases into your BPD. Vertical boxes in Process Designer represent phases and frame tasks that are correlated to the particular phase.
If you choose to use phases, a good practice is to name each with a noun. Some example conventions to name phases are:

- Approval
- Orientation
- Application processing

Screen Name: *Example process: Expense reimbursement*
Screen Number: *17*
Script:

Here is an example business process that completed the process discovery and analysis phases. The resulting documentation for the process contains the following lanes: participants, submitter, and approver. The process also contains the following phases: submission, approval, and payment. It also contains a default start and end event.
Translating the process information would yield a start to the process model BPD in IBM Process Designer similar to the one shown on the screen. Process documentation from playback zero contains more information. The new information allows authors to add BPMN elements that reflect work steps for each participant. The new elements include process flow controls.
This new information enables an author to continue the creation of the process model. The next section will cover the translation of business process documentation into key process model events, tasks, and process flow.

Screen Name: *Demonstration – Modeling teams and phases*
Screen Number: *18*
Script:
It’s now time for a demonstration. To begin the demo, click the Start Demo button.

Screen Name: *Exercise 3 – Modeling teams*
Screen Number: *19*
Script:
It is now time for a hands-on exercise. To begin the exercise, click the Exercise 3 link immediately following this unit in the Course Outline window from which you launched this unit.

Screen Name: *Topic 4: Flow objects*
Screen Number: *20*
Script:

At this stage of diagramming a business process, an author considers flow objects for the model. Flow objects in a process model are in the lane for participants because they represent either process task assignments or process controls. The commonly used types of flow objects are covered in this unit: the control flow objects. Other flow object types are covered later in this unit and in other units in this course.

Screen Name: *Events: Start events*
Screen Number: *21*
Script:

The most common control flow object for a process model is the event. Just like the definition of an event in everyday life, an event is an occurrence during a process. There are three kinds of events: Start, intermediate, and end. In the initial process model, it is important that there are start and end events represented. The focus right now is on the start and end events, and the intermediate event is covered in a different unit.

A start event is a circle that a single line encompasses. Start events trigger the initiation of the process through a manual or automatic input. Make sure to describe the start input in the properties tab documentation box that is provided for the start event.

As you can see, there are three types of start events in Process Designer: none, message, and ad hoc. The two types of start events that are most commonly used are none and message. The ad hoc start event is not covered in the next section.

Screen Name: *Events: Multiple start events*
Screen Number: *22*
Script:

Now, take a closer look at both none and message start events.

None

A circle that does not have an internal image represents a standard start event (or a none start event). In a BPD, you can have only a single standard start event. If you do try to model more than one standard start event, the process does not run and alerts you to the error. Creation of a new BPD gives by you one standard start event default.

Message

Another type of start event is message. A start event with an internal marker of an envelope represents a message start event. Sometimes you might want to start a process in another manner, yet are allowed only one normal start event per business process definition. Because of the limitation, in addition to the normal start event, you can add another type of event that is called a message start event. Message start events start a process when the process application receives an external signal.

For example, your company wants their employee onboarding process to start when a new employee record is created in the company HR system. When a new record is created, the HR system sends a message to IBM Business Process Manager. IBM captures the message through the message event and starts the follow-on events for the process.

You can have more than one message start event per business process definition.

Screen Name: *Events: End events*
Screen Number: *23*
Script:

The other default event is the end event. If a dark, thick line encompasses a circle, the circle represents an end event. End events are reached in a process when a final decision from all activities or a partial set of activities is reached.

There are four types of end events: none, message, error, and terminate. The type that is used most often is the none end event.

Screen Name: *Events: Multiple end events*
Screen Number: *24*
Script:

A none end event can also be called a standard end event. A dark circle that does not have an internal marker represents a none end event. Like the none

start event, the none end event is a default event that is provided when a business process definition is created. Unlike the none start event, it is not limited to one end event per business process definition. An author can provide as many standard end events as needed to complete the business process model. So in retrospect, a business process definition can have a single standard start event and one or more standard end events per diagram.

Screen Name: *Activity: Task*
Screen Number: *25*
Script:

Another type of flow object is the activity. An activity in a process model represents a logical unit of work that a human or a system completes during process execution.

A rectangle with rounded corners represents an activity.

Screen Name: *Activity: Task types (1 of 2)*
Screen Number: 26
Script:

There are many different types of activities, but they can be thought of in three general categories: none, task, and nested process. The none activity is devoid of any markers. The task and nested process activities have icons in either the upper left corner or the lower right to indicate what activity they are. For the first part of the model, the focus is on modeling task-type activities.

Screen Name: *Activity: Task types (2 of 2)*
Screen Number: 27
Script:

None

A rounded corner rectangle with no icons or symbols represents an activity of type none and has no implementation. This type of activity can be useful when initially modeling if an author is unsure of its implementation because of vague requirements. The author might also use it during the early stages of model analysis. As the model evolves, the activity type can be changed to something more specific. This type is not displayed by default, and you select it in the

implementation section if it is something you would like to use. Because requirements for exercises are clearly defined, this type of activity is not used.

Task activities

All task activities have an icon in the upper left corner of the activity to indicate their type.
The four types of task activities are user task, system task, decision task, and script (task).
User and system tasks are the more common types of task activities that are used for modeling. A decision task is used in the implementation course.

User tasks

An activity with a human icon in the upper left corner represents a user task. User tasks are selected if a user or human started or completed the activity. One example of a user task is an employee who is filling out an expense report and submitting it for compensation. If you drag an activity from the Designer element palette to a non-system participant lane, you automatically see a user task activity on your BPD.

System tasks

An activity with two gears in the upper left corner represents a system task. System tasks are selected if an automated system or service completes the activity. One example of a system task is payment of an expense. After an expense is approved, you might want a system to go through steps to process and automatically pay the employee. If you drag an activity from the Designer element palette to the system lane, you automatically see a system task activity.

Screen Name: *Example process: Expense reimbursement*
Screen Number: 28
Script:

Process documentation provides details on the participants in a business process and also the work they complete. It is important to note that this information is primarily captured and, in some instances, analyzed for value. However, more changes can happen after it is represented in a process model. It is also information that is work-related and not necessarily conducive to process model needs until translated. This documentation requires more comprehensive work from the process author to model process activities for participants correctly. Authors must not create a process model that is too complicated to communicate the business process effectively.

Recall one of the guidelines:

- In general, a BPD needs to be as simple an abstraction as can be modeled. A highly conceptual BPD is resilient to change.

To create the simple abstraction process model, authors go through a series of changes to the activities. When the initial process model with these refined activities and process flow is in place, BPM analysts continue to analyze the process. BPM analysts and the process owner look for process improvements, appropriate automation opportunities, and finally, a stable process model that can be implemented.

What is important to remember is that a good process model typically has an activity represent a single task that a process participant accomplishes. Modeling activities and nested processes are covered in the next section, and how an author can represent single task activities effectively in a process model.

Screen Name: *Translating business process work steps into activities*
Screen Number: *29*
Script:

Capturing the process in the discovery phase does not necessarily mean that the captured work steps are filtered into logical units of work. It depends on those analysts that capture and document the data, and the depth of their work. The better the process discovery, the easier the translation of work steps into process model activities.

Here is an example of work steps that are captured in a discovery session for the Expense Reimbursement process.

Screen Name: *How was this translation accomplished?*
Screen Number: *30*
Script:

Now look at the initial process model with the activities from the work steps. How was the translation from the work steps to process activities completed?

Screen Name: *Translation*
Screen Number: *31*
Script:

Here is how the work steps captured in process discovery translate into process model activities. One process participant completes one logical unit of work. Notice that the activities or tasks are titled by using a verb-noun statement. The work steps can be documented in each of the BPD activity properties tabs.

Screen Name: *Guidelines for modeling activities (1 of 2)*
Screen Number: *32*
Script:

When modeling activities in the Process Designer, here are some guidelines to follow:

- Remember, activities represent logical units of work that are assigned to a process participant. If not initially, then eventually activities in a BPD need to be refined into those logical units of work.
- Convert multiple concurrent work steps that are assigned to one responsible role into one activity or task.
- Use verb-noun statements to label activities.

Screen Name: *Guidelines for modeling activities (2 of 2)*
Screen Number: *33*
Script:

Here are some additional guidelines to follow:

- A process model reads best when the diagram has a top-down, left-to-right flow.
- If you are using phases in a BPD, make sure that the activities are aligned to each phase. For example, the Approval phase would naturally have the “Approve by Manager” activity in its quadrant of the model.

Screen Name: *Connecting flow objects*
Screen Number: *34*
Script:

Now that flow objects are being added to the process model, it is necessary to provide the basic flow for these activities.

This type of diagramming provides communication of how the process flows from one activity to another and who does those activities. The method to accomplish this connection of flow objects in Designer is as follows:

- Click one time to anchor on one flow object, and then drag the cursor to the next flow object.
- Click one time to connect.

- Using the rule of thumb of top-down, left-to-right flow, connect flow objects from left to right or from top to bottom on the objects. This rule helps with the simplicity of the process model and helps keep flow lines from crossing.

The early stage process model is necessary to communicate the expected flow of the process from the start event, through all activities, to the end event. As the process model is analyzed and adjusted, the process flow is modified to express the nuances of alternative flow. In the next unit, process flow is covered more comprehensively.

Screen Name: *Demonstration – Model task-type activities and events*
Screen Number: *35*
Script:
It is now time for a demonstration. To begin the demo, click the Start Demo button.

Screen Name: *Exercise 4 – Modeling task-type activities*
Screen Number: *36*
Script:

It is now time for a hands-on exercise. To begin the exercise, click the Exercise 4 link immediately following this unit in the Course Outline window from which you launched this unit.

Screen Name: *Unit 3: Creating a process model*
Screen Number: *37*
Script:
Topic 5: Nested process. Activities that are gleaned from process discovery are not always going to be tasks that effectively communicate the business process in a simple manner. To judge communication simplicity and effectiveness of the process model, many use this statement:

- Clearly and easily communicate BPDs in 5 minutes or less at any level of granularity.

Screen Name: *Activity: Nested process*
Screen Number: *38*
Script:

The goal is to achieve the effective communication of the process model. Authors must use decomposition to judge whether they are at simple abstractions of the

model with single logical units of work for activities. Decomposition is basically showing details for a business process by using a series of process model definitions that are connected as high level to child definitions.

In essence, a flow object activity is also used as a container of child definitions, which in turn can have activities that also contain child definitions. When the activities within the child definition represent logical units for work, or tasks for each activity, then decomposition is no longer necessary.

Child definitions in process models are known as nested processes. All nested process activities indicate their type in two different ways. They have a plus sign with a square symbol in the lower-right corner to indicate that they are a nested process and not a task-type activity. The other way a nested process is designated is as an activity that has a distinct outline from other activity task types.

Is decomposition always necessary for process modeling?

Decomposition is necessary if the process model is too complex to communicate the details of the business process. Decomposition is also necessary if the activities do not necessarily equal logical units of work. Authors would not need to decompose if the process documentation produced the information that led to a simplified BPD.

Screen Name: *Activity: Nested process types (1 of 2)*
Screen Number: 39
Script:

The three types of nested process activities are subprocess, linked process, and event subprocess. The first two, subprocess and linked process, are more common than the event subprocess, which is a specific case type. The focus is on the first two nested process types in this course.

Screen Name: *Activity: Nested process types (2 of 2)*
Screen Number: 40
Script:

You now take a closer look at the two nested process types: subprocess and linked process.

Subprocess

An activity with a plus sign and square in the lower-right corner and a normal single line that encompasses the activity indicates a subprocess.

This type of nested process can be thought of as a subset of the original process. Sometimes, the subprocess is called an embedded nested process. Authors cannot reuse this type of nested process, and they think of this reuse when authors choose this type of nested process. If the nested process might be reused in another process, choose a linked process instead.

Authors think of this nested process type as hiding several activities from view. You can achieve your goal of communicating your process in 5 minutes or less by using nested processes.

Linked process

A linked process is an activity with a plus sign and square in the lower-right corner. It also contains a dark bold line that outlines the activity indicates a linked process.

This type of process might be thought of as distinct or separate from the original process, unlike the subprocess. You can reuse this type of process in many different parent processes. An example of reuse is a legal review that can be used in several different insurance claims processes. If your legal review is the same in an auto insurance claim and in a property insurance claim, you would want to choose a linked process.

If you drag an existing BPD onto the canvas of another BPD, it results in a linked process.

Screen Name: *Subprocess example (1 of 2)*
Screen Number: *41*
Script:

Here is an example of a subprocess in Designer.

Notice that the approval phase had an update to the tasks for the approver participants. Instead of two approval tasks, there is now one nested process activity to represent the logical unit of work: Approve Expense Report.

After consulting with the business, it is determined that this process is a unique process and it will not be reused. For this reason, this process is a subprocess type of activity. After you rename the activity, select the activity and choose subprocess from the Implementation section.

Screen Name: *Subprocess example (2 of 2)*
Screen Number: *42*
Script:

When you double-click the activity, the subprocess displays. Here, authors create the model for the subprocess, copying the former activities. Notice there is a breadcrumb trail at the top so an author can go back to the top-level process.

In the second BPD, the participants were narrowed down to two. Notice that the system lane does not exist in the nested process.

Screen Name: *Demonstration – Nested process and decomposition*
Screen Number: *43*
Script:
It is now time for a demonstration. To begin the demo, click the Start Demo button.

Screen Name: *Exercise 5 – Creating nested processes and decomposing in a BPD*
Screen Number: *44*
Script:

It is now time for a hands-on exercise. To begin the exercise, click the Exercise 4 link immediately following this unit in the Course Outline window from which you launched this unit.

Screen Name: *Unit summary*
Screen Number: *45*
Script:

Having completed this unit, you should be able to:

- List and describe the core notation elements that are used in IBM Process Designer
- Examine a defined workflow from detailed process requirements and identify the interrelated process activities and the roles that are responsible for completing them
- Decompose activities into processes and nested processes that contain process tasks
- Create a business process definition (BPD) from the process and nested process tasks and responsible roles

---

Unit 4: Defining process flow

Screen Name: *Unit introduction*
Screen Number: *1*
Script:
Welcome to Unit 4: Defining process flow.
This unit is about process flow and what you can add to your model to affect the flow of the process. Until this unit, the focus of translating the process requirements into a process model was on the expected process flow. The focus was on the happy path of the process. However, most real-life business processes cannot follow a single path from start to end. To continue to translate the documented requirements, conditions and exceptions must be modeled. These conditions and exceptions create alternative flows in your process definitions. The order in which activities are done is defined for both the expected and alternative process flows.

Screen Name: *Unit 4: Defining process flow*
Screen Number: *2*
Script:
After completing this unit, you should be able to:

- Describe process sequence flow and the runtime use of process tokens
- List and describe the gateways as they are used in IBM Process Designer
- Explain how to evaluate conditions for a BPD gateway
- Model gateways in a BPD
- List and describe intermediate event types that are used in IBM Process Designer
- Model a business process escalation path with an attached timer intermediate event

Screen Name: *Unit 4: Defining process flow*
Screen Number: *3*
Script:
The first topic is about process flow.
Comprehensive process models have one thing in common; they communicate process flow well. Process flow encompasses both the normal, expected process path to completion, and alternative process paths that might occur with different process conditions or business rules. It is important to understand what types of sequence flow exist in process modeling and how to implement gateways. This understanding helps you to communicate both kinds of process flows in the process model.

Screen Name: *Normal sequence flow*
Screen Number: *4*
Script:
Process flow is implemented in Business Process Manager with sequence flow. Sequence flow is the element that represents the path of your process. It is a

connecting object that is drawn from one step to the next and defines flow from one step to the next. There are three types of sequence flow. The first type of sequence flow is normal sequence flow. Normal sequence flow is used to describe how a process moves from one step to the next in a normal, uninterrupted fashion. A normal plain arrow represents normal sequence flow. These flows are unidirectional lines between the steps.

Screen Name: *Conditional sequence flow*
Screen Number: *5*
Script:
The next type of sequence flow is conditional sequence flow. The line type is an arrow with a mini diamond attached to the end of the line. Conditional sequence flows are associated with gateways most the time. From a pure modeling perspective, the diamond visually expresses that there is a condition on the line. During the process, the condition is evaluated and if it is true, that sequence flow is taken. If the condition is not true, that sequence flow is not taken.

Screen Name: *Default sequence flow*
Screen Number: *6*
Script:

The last sequence flow is default sequence flow. This type of flow is typically associated with gateways, and an arrow with a hash on the line represents this flow. This concept is important. Default sequence flow is only ever used with conditional sequence flow. If you have even one conditional flow, you must have a default sequence flow. You can use a single default flow only in association with a set of conditional sequence flows. So, in a set of sequence flows coming from a step, you can have one to many conditional sequence flows. But, there is only one default sequence flow.

Screen Name: *Unit 4: Defining process flow*
Screen Number: *7*
Script:
Tokens are used to describe how the process flows when the process is run. The indication of a token on a step identifies the location of active processing steps of that business process.

Screen Name: *What is a token?*
Screen Number: *8*
Script:

This image is a process token. A token is a theoretical object that is used to indicate how a process moves from one step to the next. In Business Process

Manager, you cannot access a token to do actions on it. It is one of the reasons why it is called a theoretical object. It is purely visual. It can be thought of as a "you are here" marker.

Screen Name: *Traverses the flow objects*
Screen Number: *9*
Script:

This image shows a token that is moving from one step to the next in a normal uninterrupted fashion. The lighter tokens show where the token was previously, just for representation purposes. When you use Business Process Manager, you see only a red token, or where the token currently is. In the course diagrams, the lighter pink tokens are there to show you where the token was before.

Screen Name: *Diverted through alternative paths*
Screen Number: *10*
Script:

The token can be diverted through one or many alternative paths.

Screen Name: *Split into parallel paths*
Screen Number: *11*
Script:

The tokens can split to go to more than one step at a time and run tasks or nested processes at the same time.

Screen Name: *Unit 4: Defining process flow*
Screen Number: *12*
Script:
Topic 3: About gateways. Often, a gateway is used to control flow of a process.

Screen Name: *Represented as a diamond*
Screen Number: *13*
Script:
A gateway is always drawn as a diamond shape. Some gateways have internal markers inside the diamonds. The gateway looks at the conditions of the process data. The diamond represents a point in the process where a question is asked.

Screen Name: *Can be thought of as a question*
Screen Number: *14*
Script:
Often, a gateway is representative of a question that is asked at a particular point in a process.

Screen Name: *Has a defined set of alternative answers*
Screen Number: *15*
Script:
Think of the sequence flows that leave the gateways as different answers to the question. If the question is answered correctly, the process flow token proceeds down that line.

Screen Name: *Two distinct modes: Split*
Screen Number: *16*
Script:

Gateways can have two distinct modes. The way these modes are defined depends on how many flows enter and exit the gateway (incoming and outgoing sequence flow). In this example, there are a single incoming sequence flow and multiple outgoing sequence flows. This mode of gateway can behave as a split.

Screen Name: *Two distinct modes: Join*
Screen Number: *17*
Script:

If the diagram has multiple incoming sequence flows and one outgoing sequence flow, it is known as a join. You can also draw a gateway with multiple incoming and multiple outgoing flows. It is valid, but it is not something that is generally practiced.

Screen Name: *Exclusive: Diamond shape with no internal marker*
Screen Number: *18*
Script:

There are three types of gateways. The first type of gateway is an exclusive gateway. An exclusive gateway is represented as a plain diamond. Exclusive gateways do not act as splits and joins in Business Process Manager. An exclusive gateway works only in split mode.

Screen Name: *Exclusive: One or more outgoing conditional sequence flows*
Screen Number: *19*
Script:

Here are the outgoing flows for the exclusive gateway. They are conditional sequence flows. The process takes only one of these conditional flows. Exclusive gateways need to be modeled so that only one outgoing sequence flow is true. After the condition is met for any one of the sequence flows, the token takes that particular flow line.

Screen Name: *Exclusive: Default sequence flow*
Screen Number: *20*
Script:

Remember that every gateway with one or more conditional sequence flows also has a default sequence flow. Here there is one, and only one, default sequence flow with two conditional flows. If none of the conditions on the conditional sequence flow are true, the process takes the default sequence flow.

Screen Name: *Process narrative*
Screen Number: *21*
Script:

This example is an example of an exclusive gateway. Here is the process narrative for the submit auto damage claim process. If the claim amount is less than 1,000 dollars, conduct a small claims review. If the claim amount is 1000 - 5000 dollars, conduct a standard claims review. If the claim amount is greater than 5000 dollars, conduct a fraud claims review.

Screen Name: *Submit auto claim example (1 of 6)*
Screen Number: *22*
Script:

Only one of these conditions can be true at a time, so the gateway to use is an exclusive gateway. Here is the modeled process. The gateway is labeled with a question, and the outgoing lines answer the question.
Notice that the line labels are more descriptive than yes or no. The process can be read in one glance. You do not have to read the gateway question to determine the meaning of the lines.

Screen Name: *Submit auto claim example (2 of 6)*
Screen Number: *23*
Script:

The token starts on the first activity, Submit auto claim.

Screen Name: *Submit auto claim example (3 of 6)*
Screen Number: *24*
Script:

The token then moves down the normal sequence flow to the gateway. Remember that tokens traverse every step, and here the token is on the gateway.

Screen Name: *Submit auto claim example (4 of 6)*
Screen Number: *25*
Script:

The damage was 700 dollars. The first condition is true, and the token travels through the top conditional flow because the damage is under 1000 dollars.

Screen Name: *Submit auto claim example (5 of 6)*
Screen Number: *26*
Script:

What if the damage is 2000 dollars? The top conditional flow is not true. Because the flow is not true, the middle line is checked for truth. The middle conditional flow is true, so the token travels through the middle conditional sequence flow.

Screen Name: *Submit auto claim example (6 of 6)*
Screen Number: *27*
Script:

What if the data does not meet either of the conditional flows? The claim is 8000 dollars. The claim is not under 1000 dollars and is not in the range of 1000–5000 dollars. If none of the conditions on the conditional flows are true, the token travels the default sequence flow.
Notice that there is a label on the default flow to specify what is happening. Despite the label, the default flow line is not evaluated to see whether it is true or false. It is taken if the others are not true. This information is important to remember when you model the gateway. It is a good idea to preserve one of your process requirements for the default flow.

Screen Name: *Inclusive: Diamond shape with an internal circle*
Screen Number: *28*
Script:

The second type of gateway that is used in Business Process Manager is an inclusive gateway. This gateway is a diamond with a circle inside the diamond as the internal marker. When you see this symbol, you have an inclusive gateway. Inclusive gateways can function in the split or join mode in Business Process Manager.

Screen Name: *Inclusive split: One or more outgoing conditional sequence flows*
Screen Number: *29*
Script:

With an inclusive gateway, you can have many conditional flows. The difference between an inclusive gateway and an exclusive gateway is that more than one conditional flow can be true at a time.

Screen Name: *Inclusive split: Default sequence flow*
Screen Number: *30*
Script:

Because there are outgoing conditional flows, there must be one default flow line. The default sequence flow is taken when none of the conditions on the conditional sequence flows are true.

Screen Name: *Process narrative*
Screen Number: *31*
Script:

Here is an example of a business process with a need for an inclusive gateway. This process is for when damage occurs to an apartment building. When damage occurs to the building, the facilities manager must submit a damage report. If fire causes the damage, the fire department must be notified. If the amount of damage is greater than 5000 dollars, the insurance agent must be notified. There are two rules in this process. Both can be true, or one can be true, or none can be true. Regardless of the two conditions, any time the building is damaged and a report is submitted, the building manager must always be notified.

Screen Name: *Damage report example (1 of 8)*
Screen Number: *32*
Script:

Here is a proposed process model from that set of requirements.

Screen Name: *Damage report example (2 of 8)*
Screen Number: *33*
Script:

The token starts on Submit damage report as expected.

Screen Name: *Damage report example (3 of 8)*
Screen Number: *34*
Script:

The token moves to the next step down the normal flow: the gateway.

Screen Name: *Damage report example (4 of 8)*
Screen Number: *35*
Script:

Fire caused the damage, and the damage is 3000 dollars. Unlike the exclusive gateway, all of the conditional flows are checked or evaluated because it is an inclusive gateway. Here the top conditional flow is checked and it is true, so a

token proceeds to Notify fire department. The middle conditional flow is checked, but is not true.

Screen Name: *Damage report example (5 of 8)*
Screen Number: *36*
Script:

Another scenario might be that fire causes the damage, and the damage is 8000 dollars. In this case, the top condition is checked and since it is true, the token proceeds down the conditional flow to Notify fire department. The next condition, “the damage is over 5000 dollars”, is checked. Since the damage is 8000 dollars, the middle condition is also true, and another token proceeds down the conditional flow to Notify insurance agent.

Screen Name: *Damage report example (6 of 8)*
Screen Number: *37*
Script:

Another scenario might be that a car crashed into the lobby. The damage is 20,000 dollars, and fire did not cause the damage. The top condition was evaluated, and it was not true since the car caused the damage. There is no token. The middle condition was evaluated, and because the damage is 20,000 dollars, which is over 5000 dollars, this condition is true. A token proceeds down the conditional flow to Notify insurance agent.
All scenarios prove how each condition can be individually true, or both conditions can be true concurrently, and how the inclusive gateway handles each scenario.

Screen Name: *Damage report example (7 of 8)*
Screen Number: *38*
Script:

The building has a broken window, fire did not cause the damage, and the damage is only 500 dollars. This type of situation is really the only time that the default sequence flow is taken. It is only traversed when all the conditions are not true.
This model, however, does not meet the requirement that the building manager is always informed when damage is reported. In this model, the building manager is informed only when the top and middle conditions are not met.

Screen Name: *Damage report example (8 of 8)*
Screen Number: *39*
Script:

This model is a better diagram. Now, the building manager is always notified. But if both the top and middle conditions are both true, the building manager is notified twice for the same incident.

Screen Name: *Inclusive join example*
Screen Number: *40*
Script:

Recall the join mode of a gateway. Inclusive gateways can also function as joins. An inclusive gateway can be a split and a join. The inclusive join might receive two tokens (one from Notify fire department and one from Notify insurance agent). If so, the join passes only one token to Notify building manager. The building manager is now notified only one time.
This model now meets all of the requirements.

Screen Name: *Parallel gateway: Diamond with an internal plus*
Screen Number: *41*
Script:

The third type of gateway is known as a parallel gateway. A diamond with a plus as the internal marker represents a parallel gateway. Parallel gateways can also act in split and join modes.

Screen Name: *Parallel split: No conditional or default flows*
Screen Number: *42*
Script:

The parallel gateway is different from the other two gateways. Every outgoing flow is a normal sequence flow. This situation means that the outgoing flow does not require meeting a condition or a default flow to mitigate the conditions that are not met. Every outgoing flow from the parallel gateway gets its own token.

Screen Name: *Process narrative*
Screen Number: *43*
Script:
Here is an example process in need of a parallel gateway. This process is part of an onboarding process for an HR team. After an employee is hired, here are some of the tasks they need to complete on their first day. On the first day of employment, employees must complete the HR new hire forms first. Then, they must apply for a security badge, requisition a computer, and apply for a network ID and email address.
In this process, no order is specified on the last three tasks. There is an opportunity here to run tasks at the same time. Anytime that you want to run more than one task consecutively in an uncontrolled manner, you can use a parallel split gateway.

Screen Name: *New hire on-boarding example (1 of 4)*
Screen Number: *44*
Script:
Here is a proposed model.

Screen Name: *New hire on-boarding example (2 of 4)*
Screen Number: *45*
Script:
How do the tokens proceed? The token starts on the complete new hire forms activity.

Screen Name: *New hire on-boarding example (3 of 4)*
Screen Number: *46*
Script:
The token proceeds to the gateway.

Screen Name: *New hire on-boarding example (4 of 4)*
Screen Number: *47*
Script:
Because this gateway is a parallel split, every outgoing path gets a token. Here, there are three outgoing normal flows, so there are three tokens.

Screen Name: *Parallel join*
Screen Number: *48*
Script:
If you need to join the three tokens that are coming from the activities, use a parallel join here. Sometimes you need to be careful when you use parallel joins. A parallel join waits for a token from every incoming sequence flow. You cannot time it out or express a condition on a parallel join to change its behavior. For example, in this process, a parallel join waits for three tokens, one from each activity before the token moves to the next activity.

Several processes might require this behavior. In the process here, maybe the next step is to set up your desk. Maybe you do not want the participant to set up their desk until all three of the previous tasks are accomplished. In this case, you use a parallel join.

Screen Name: *Name the gateway*
Screen Number: *49*
Script:
There are three types of gateways: exclusive, inclusive, and parallel. Inclusive and parallel are the only gateways that can be used in a split and join fashion. Exclusive gateways cannot be used the same way. In Business Process Manager, an exclusive gateway cannot be used to join incoming flows.
Here is an activity to look at gateways that are used in combinations. This diagram needs to work as designed and expected, meaning that a token starts at the left end on the gateway in the upper right.
Everything here is accurately represented except for the flow. Do not pay attention to the flow; look at the symbols and write down your answer. When you are done, move to the next screen and roll over the question marks to see whether you are correct.

Screen Name: *Name the gateway solution*
Screen Number: *50*
Script:
The token starts here. Which gateway is this first one?
It is a parallel split, so there is a token here and a token there. There are two now. Follow the top token. It traverses this flow and encounters another parallel split, so here it splits into two and there are two tokens at the top. There are a total of three tokens in the process. Look at this first question mark. This gateway is a join of some type because of the number of incoming versus the number of outgoing sequence flows. You need a join here to join the top two tokens.
Look at the bottom token. The bottom token landed on a gateway. This gateway is an exclusive gateway; it is correctly marked. On an exclusive gateway, this token can traverse the top or the right outgoing flows. Perhaps it goes to the right.
If you use a parallel join at the top on the process, it waits for tokens from all of the incoming lines. What if the exclusive token went to the right? It would stall the process flow.
Use an inclusive join so the process flow can continue, no matter which way the token goes on the exclusive gateway. What is the other gateway? This gateway is also an inclusive join for the same reason. It is because you cannot guarantee tokens on the two incoming sequence flows. You might not be able to guarantee a token proceeds down an incoming flow. If you need to join tokens, you might want to use an inclusive join instead of a parallel join.

Screen Name: *Evaluating conditions: Decision logic in the outgoing sequence flow*
Screen Number: *51*
Script:
Normally, when you are evaluating a condition, some work must be done, such as submitting an order. In the example that is provided, if the evaluation of the "order is valid" is parallel, your outgoing lines can do the evaluation work.

Screen Name: *Evaluating conditions: Externalized exclusive logic*
Screen Number: *52*
Script:
However, much the time, the answer is not so parallel. Often, you put another activity in front of the gateway to break it apart. The process did not change. The logic moved outside of the gateway. An activity is used to represent complex business rules, in this case.
Business rules often change, and it is important to make sure that your process model can withstand these sorts of changes. This reason is why the logic of the gateway is modeled outside of the gateway. It then sends the exclusive data to the gateway (as shown in the diagram).

Screen Name: *Demonstration - Gateways*
Screen Number: *53*
Script:
It is now time for a demonstration. To begin the demo, click the Start Demo button.

Screen Name: *Exercise 6 – Modeling gateways in a BPD*
Screen Number: *54*
Script:

Exercise 6: Modeling gateways in a BPD

Screen Name: *Unit 4: Defining process flow*
Screen Number: *55*
Script:
Topic 4: Intermediate events. If an event takes place between a start and end event in the BPD, it is known as an intermediate event. An overview of the type of intermediate events that used in Process Designer is covered in the next topic.

Screen Name: *Intermediate events*
Screen Number: *56*
Script:

The intermediate event is designated by a double-lined circle and the internal marker specifies the type of intermediate event that is taking place in the model. In Process Designer, there are four types of sequence flow intermediate events:

- Message intermediate event that sends or receives a message
- Content intermediate event
- Timer intermediate event
- Tracking intermediate event

These events are free standing and are placed in-line with attached flow in the process. It differs from the boundary or attached intermediate events, which are associated with an activity on the BPD. The four types of boundary intermediate events are:

- Error
- Message
- Content
- Timer

Screen Name: *Intermediate events: Timer*
Screen Number: *57*
Script:

Timer intermediate events are used to model delays in your BPDs. Boundary or attached timer intermediate events are used for escalation paths. Using a timer intermediate event, an author can specify a time interval for the process to pause after an activity is completed or before and activity is assigned to a user.

Screen Name: *Intermediate event: Message*
Screen Number: *58*
Script:
Message intermediate events are used to model a message that is sent or received while a process is running. The lighter envelope indicates that a message is received, while the darker envelope indicates that a message is sent.

Screen Name: *Intermediate event: Error*
Screen Number: *59*
Script:

Error intermediate events are used to catch process execution exceptions and handle exceptions with an error handler activity or further process flow.

Screen Name: *Intermediate event: Content*
Screen Number: *60*
Script:
Content intermediate events provide process modelers the ability to react to events on an Enterprise Content Management system (ECM). Use the content intermediate message event if a document sent to an Enterprise Content Manager affects the flow of a process.

Screen Name: *Intermediate event: Tracking*
Screen Number: *61*
Script:

Tracking intermediate events are used to indicate a point in a service when Process Designer captures the runtime data for reporting purposes. This particular event is an IBM specific intermediate event, unique to use in Business Process Manager.

Screen Name: *Attached and sequence flow intermediate events*
Screen Number: *62*
Script:
All intermediate events behave the same way; they respond to a specific event at a specific point and time in the BPD. Intermediate events can be implemented two ways:

- Sequence flow: an intermediate event that is placed at a point in the process flow and affects the flow of the BPD
- Attached: an intermediate event that is attached to the activity and affects the activity directly

---

Screen Name: *Sequence flow intermediate event*
Screen Number: *63*
Script:

The following is an example of the sequence flow intermediate event. Specifically, an intermediate event in the sequence flow pauses the process flow until the event occurs. All intermediate events in sequence flow work this way except for the tracking event.

Screen Name: *Process stops until intermediate event occurs*
Screen Number: *64*
Script:

The process encounters an event in the sequence flow (for example, an incoming message) and stops until the event occurs.

Screen Name: *Process continues on sequence flow when intermediate event completes*
Screen Number: *65*
Script:

When the specified event occurs, the process flow continues along normal sequence flow.

Screen Name: *A common requirement (1 of 5)*
Screen Number: *66*
Script:

This diagram is a common usage pattern for an intermediate message event that is modeled in the sequence flow of the BPD. This process involves processing an order. To process the order, the process must first check to see that the item ordered is in stock.

Screen Name: *A common requirement (2 of 5)*
Screen Number: *67*
Script:

After an item is ordered, the first task in the business process is to check the inventory for the item.

Screen Name: *A common requirement (3 of 5)*
Screen Number: *68*
Script:

The evaluation of the inventory check moves the token to the next flow object, an exclusive gateway. Here, one of the two sequence flows is selected based on the condition met: “in stock”. If the item is “in stock”, then the sequence flow moves the token to the next task: process the order that the customer placed.

Screen Name: *A common requirement (4 of 5)*

Screen Number: *69*
Script:

In this case, the condition was not met that the item was “in stock”, and the default sequence flow was taken. Now the process must await a response from the inventory warehouse that the item is restocked at the fulfillment warehouse. The process cannot move to the next task until the intermediate message event triggers the go-ahead that the event occurred.

Screen Name: *A common requirement (5 of 5)*
Screen Number: *70*
Script:

The message is received that the inventory at the fulfillment warehouse was replenished. Then, the process can move to the next task in the sequence flow. The customer order is processed and shipped.

Screen Name: *Attached intermediate event*
Screen Number: *71*
Script:

As noted earlier, intermediate events can also be attached to the boundary of an activity.

Screen Name: *Process flow reaches an activity*
Screen Number: *72*
Script:

When the normal process flow reaches an activity with an attached intermediate event, the attached intermediate event generates a separate token.

Screen Name: *Attached intermediate event generates a separate token*
Screen Number: *73*
Script:

The attached intermediate event is only active while the activity to which it is attached is active. If the specified event does not occur before the activity completes, the token on the attached intermediate event ends.

If the specified event occurs during the execution of the activity, the outgoing flow from the attached intermediate event can react in several ways.

Screen Name: *Attached intermediate event can create a parallel flow*
Screen Number: *74*
Script:

One way the attached intermediate event can react is to create a parallel process flow. It is accomplished by releasing its token, and leaving the token active on the attached activity.

Screen Name: *Attached intermediate event can close an activity*
Screen Number: *75*
Script:

Another way the attached intermediate event reacts when the event occurs is to close the attached activity, and release its token down an alternative process flow. For this case, the attached intermediate event must be defined to react this way.

Screen Name: *Attached intermediate event example (1 of 4)*
Screen Number: *76*
Script:

This diagram is an example model for processing a customer order. In this case, the intermediate event is an attached intermediate event that is attached to the boundary of the Process Order activity.

Screen Name: *Attached intermediate event example (2 of 4)*
Screen Number: *77*
Script:

After the customer order is ready to be processed, the task to complete is to process the order. The token sits on this activity until the task is completed.

Screen Name: *Attached intermediate event example (3 of 4)*
Screen Number: *78*
Script:

While the first token waits for the processing of the order, a second token is created for an attached message intermediate event. While the order is processed, there is a chance the customer or someone else cancels the order

because of extenuating circumstances. Perhaps they changed their minds, or the wrong order was placed.

Screen Name: *Attached intermediate event example (4 of 4)*
Screen Number: *79*
Script:

The attached message intermediate event is designed to close the activity, “Process Order”, if the message was received that the order is canceled. Only the token that is generated for the attached intermediate message event is then released on the alternative flow. Then, the task to cancel the order from the order fulfillment database is created.

Screen Name: *Attached intermediate event is active only when the activity it is attached to is active*
Screen Number: *80*
Script:

What would happen to the attached intermediate event if an activity such as “process order” were completed? Any attached intermediate event is active only when the activity it is attached to is active.

Screen Name: *Unit 4: Defining process flow*
Screen Number: *81*
Script:
Topic 5: A way to model an escalation. In some cases, the process interactions in the model represent delays in the deliverable, notification, or escalation added to tasks that can get stuck. Prescribed delays or interactions that deal with deadlines can be represented with a timer intermediate event in the BPD. Modeling an escalation is covered in the next topic.

Screen Name: *A way to model an escalation*
Screen Number: *82*
Script:

Some business requirements that an author might encounter include statements such as “complete this task by a specified date or time”. If the task or activity is not completed, the model must indicate what would happen. The model includes to whom the task or activity escalates and when the task or activity escalates. Modeling escalations is a reaction to the provided visibility for your business process and implements controls to manage process cycle times. Escalations also communicate service level agreements for the process tasks that are met.

In Process Designer, one way to model an escalation is with an attached timer intermediate event. An attached timer intermediate event is triggered if an activity takes longer to complete than a specified time. Then, the process follows the path from the attached timer intermediate event to an activity that is an escalation activity.

Screen Name: *Timer intermediate event*
Screen Number: *83*
Script:
The timer intermediate event allows a process to wait explicitly or react to the passing of time.
The time interval can be based on system time, a deadline, or a custom time.
An example is the loan application process to approve loan activity. If the approver does not approve a loan within 10 days, the loan approval is escalated to a loan manager.

Screen Name: *Demonstration – Intermediate events*
Screen Number: *84*
Script:
It is now time for a demonstration. To begin the demo, click the Start Demo button.

Screen Name: *Exercise 7- Modeling timer intermediate events*
Screen Number: *85*
Script:
It is now time for a hands-on exercise. To begin the exercise, click the Exercise 7 link immediately following this unit in the Course Outline window from which you launched this unit.

Screen Name: *Unit summary*
Screen Number: *86*
Script:

Having completed this unit, you should be able to:

- Describe process sequence flow and the runtime use of process tokens
- List and describe the gateways as they are used in IBM Process Designer
- Explain how to evaluate conditions for a BPD gateway
- Model gateways in a BPD
- List and describe intermediate event types that are used in IBM Process Designer
- Model a business process escalation path with an attached timer intermediate event

Unit 5: Playback 0: Validate the process model with playback zero

Screen Name: *Unit introduction*
Screen Number: *1*
Script:
Playback zero is an important phase in BPM project development because it is where gains in business process efficiency and effectiveness are defined, adjusted, and modeled. In this phase, a business process is initially defined in its current state. It is modeled to begin the project development journey, and then refined through a comprehensive analysis of the business process.
This unit focuses on the specific goals for playback zero and how a process model reaches consensus in the validation session.

Screen Name: *Unit 5: Validating the process model with playback 0*
Screen Number: *2*
Script:
After completing this unit, you should be able to:

- Describe the playback zero validation goals and requirements
- Validate that a process model met playback zero goals and requirements

---

Screen Name: *Unit 5: Validating the process model with playback 0*
Screen Number: *3*
Script:
Topic 1: Playback zero validation phase. The basics of playback zero are covered in the next topic.

Screen Name: *Playback 0 validation*
Screen Number: *4*
Script:
In playback zero, documentation and process analysis set the correct framework for:

- The process model creation
- Process automation
- Added value process activities to gain efficiencies
- Visibility
- Effectiveness

The entire playback zero phase typically takes one to three weeks to complete.

Screen Name: *How validation works*
Screen Number: *5*
Script:

The work that is done in playback zero is reflected in the adjusted process model. It is the business process that you want, and now it can be implemented as a process application. To ensure that the business process is the right candidate to implement, stakeholders must reach a consensus that playback zero reached the final goal. This consensus is known as playback zero validation.

Playback zero validation is accomplished through a review session with all business stakeholders, business users, and the BPM development team that is attending. The validation leads to the switch from analytical modeling to executable modeling. If the validation is complete and the process model is approved, then any additional refinement to process requirements can continue. It flows through the framework of the next series of implementation playbacks in process implementation.

There are situations when a validation playback session discovers requirements that were missed or new requirements are introduced. This discovery is not uncommon in BPM because requirements change during projects. This change is why a process model is agile and why BPM provides the best system to manage change that results in continuous process improvement.

Screen Name: *Unit 5: Validating the process model with playback 0*
Screen Number: *6*
Script:
Topic 2: Reaching consensus on the process model. During the playback zero stage of project development, multiple playback sessions are held. In fact, there is typically a themed approach for these sessions within the three-week time frame. The BPM team can establish the best course of action to take for playbacks within the playback zero development. The strategy to handle each of these themed playbacks is through iteration.

These sessions take 60 – 90 minutes and must include important decision makers for the specific playback achievement that is reached. Some examples of themed playbacks for playback zero include:

- Process discovery playback
- Initial (As-Is) process model or discovery map playback
- Conceptual design (process application or report mockup designs) playback
- Measurement and visibility playback

- Validation playback

None of the listed themes are meant to be the exact themes a BPM team must use. The themes of the playbacks vary from organization to organization. However, make sure the themes match work that is done during this stage of project development. What is important is the last and final playback session: validation of the business process and process model.

Screen Name: *Validation goals*
Screen Number: *7*
Script:
The BPM team is ready to validate the business process along with the business groups, such as stakeholders and system users. The team produced the final process model that is ready for implementation.
Process analysis, process adjustment, and process modeling have all come together. The short three-week cycle enabled this final session for this stage of project development. This stage is when consensus is reached on:

- The process model is ready for implementation
- No requirements are lost in the translation effort
- If business stakeholders still express requirements, evaluate the best roadmap to implement those requirements into the model. Evaluate up to and including before validation approval is obtained

Playback zero validation is completed with the BPD inside the Process Designer tool. If everyone in the room can view the process, any tool can be used to show the process. The objective is to review all process flow, normal, and conditional, and model specifics, like participants, activities, and process control.

Screen Name: *Exercise 8 – Validating the process model*
Screen Number: *8*
Script:
It is now time for a hands-on exercise. To begin the exercise, click the Exercise 8 link immediately following this unit in the Course Outline window from which you launched this unit.

Screen Name: *Unit summary*
Screen Number: *9*
Script:

Having completed this unit, you should be able to:

- Describe the playback zero validation goals and requirements
- Validate that a process model met playback zero goals and requirements

Unit 6: IBM Business Process Manager architecture and toolkits

Screen Name: *Unit introduction*
Screen Number: *1*
Script:

IBM Business Process Manager is a comprehensive and consumable business process management platform. You can use Business Process Manager to view and manage your business processes. Business Process Manager includes tools and execution for process design, monitoring, and optimization. Business Process Manager is designed so that process owners and business users can directly improve their business processes.

Business Process Manager is a highly integrated environment, and it scales smoothly and easily from an initial project to an enterprise-wide program. IBM Business Process Manager is easy to deploy and used as-is, or it can be used in a customized configuration.

Business Process Manager provides rapid time-to-value and improved user productivity. This unit covers the basic aspects of implementing a process within Business Process Manager.

Screen Name: *Unit 6: IBM Business Process Manager architecture and toolkits*
Screen Number: *2*
Script:

After completing this unit, you should be able to:

- Describe IBM Business Process Manager product components
- Describe the relationship between the Process Center and the runtime environments or Process Servers
- Identify the process server and the performance data warehouse
- Create toolkits for easy reuse of assets

Screen Name: *Unit 6: IBM Business Process Manager architecture and toolkits*
Screen Number: *3*
Script:

This unit contains the following topics:

- IBM Business Process Manager Standard architecture overview

- Installing process applications on a process server
- Toolkits overview

The first unit is the IBM Business Process Manager Standard architecture overview. This unit provides a technical overview of IBM Business Process Manager. It starts with the components of each environment and scaling to the components of a typical deployment configuration. By identifying and understanding interactions of key components of the architecture, authors can successfully implement the business processes.

The first topic provides an overview of the Business Process Manager configuration. To learn more about server configuration and administration, IBM Education offers more in-depth classes on IBM Business Process Manager Administration and WebSphere administration.

Screen Name: *IBM Business Process Manager product components*
Screen Number: *4*
Script:

This graphic is a representation of the topology of a typical IBM Business Process Manager installation. The unique design environment of IBM Business Process Manager includes a central repository that is called the Process Center. You can see the Process Servers surround the Process Center. The Process Center is where process applications are created and stored. The runtime environments (also called Process Servers) are where those process applications are tested and production processes are created and run.

Screen Name: *The Process Center*
Screen Number: *5*
Script:

The Process Center is where the process application design occurs. Process applications are also managed and run from the Process Center. The Process Center provides authors and architects the tools to organize and govern all the artifacts that are created as part of a BPM program.

The Process Center is also where toolkits, BPDs, rules, integrations, and services are created. The Process Center contains many unique components that support process development. The Process Center is known as the development environment because process authors connect to it when they create process applications.

The Process Center consists of a process server component and a performance data warehouse.

Screen Name: *The process server*
Screen Number: *6*
Script:

The process server that is contained inside the Process Center manages and runs all business process definitions or models in the development environment. The process server also stores the version and history for process development and deployment. The Process Server is an application server that runs on WebSphere. It can run on a single node or clustered to meet the performance needs of the environment as configured by WebSphere.

Screen Name: *The process Server components*
Screen Number: *7*
Script:

Inside of the Process Server component, you find a BPD engine, service engine, and event manager. The Process Server provides the workflow facilities, such as task management, routing, and simulation.

Screen Name: *The event manager*
Screen Number: *8*
Script:

The event manager is the part of the Process Server that handles event scheduling and queuing. For example, when the Process Server receives an event, that event becomes a job in the event manager. Each job in the event manager is routed through a Scheduler, which schedules and tracks the execution of its assigned jobs.
Process components are stored and shared in the process server repository. The server has extensive levels of caching to optimize runtime performance. Process execution is dynamic, and it evaluates each step and transition within the BPD in real time. The Event Manager is a stateless machine. It does not know what happened before the current step, and it does not know what happens later in the process. It processes only the current step.
The event manager is the heart of the Process Server execution abilities. It can be started and stopped through an administration console.

Screen Name: *The performance data warehouse*
Screen Number: *9*
Script:

The performance data warehouse, or business performance data warehouse, collects performance data as processes are run in the development environment. This data collection allows stakeholders to examine reports and data during development and before a process is tested or put into production.
Similar to the Process Server, the Performance Data Warehouse is an application server that runs on WebSphere. It can run on a single node or clustered to meet the performance needs of the environment as configured by WebSphere.

Screen Name: *The Performance Data Warehouse*
Screen Number: *10*
Script:

The Performance Data Warehouse uses the process model to correlate the business events in real time. It also aggregates raw performance data into a single database view for reporting and auditing. Some reporting capabilities exist inside of IBM Business Process Manager. Other reporting packages and platforms can also integrate with the Performance Data Warehouse to extract process metrics for their own reporting needs.

Screen Name: The *Process Server*
Screen Number: *11*
Script:

Each Business Process Manager Process Server contains resources for running processes that are installed from the Process Center. The Process Servers also contain the same two parts as the Process Center: a process server and a performance data warehouse.
You can run processes in the integrated process server within the Process Center as you build them. When you are ready, you can install and run those same processes on a Process Server.

Screen Name: *The Process Server: Performance Data Warehouse*
Screen Number: *12*
Script:

Similar to the Process Center, each Process Server, or runtime environment, contains a process server component. This Process Server runs the process applications that are installed from the Process Center repository, and a Performance Data Warehouse. The Process Center is just a unique Process Server that designs process applications. It also has the added capabilities to deploy those assets to the Process Servers.

***Screen Name: The** Process Server: Multiple Process Servers*
***Screen Number:** 13*
Script:

There can be a number of Process Servers within an enterprise, depending on business need. Each environment runs independently from each other. Each environment can have unique instances that are run in the environment. Each environment can connect to different end points for integrations, and can be customized as necessary for that particular environment. Typically these environments are designated as development (the Process Center), staging, test, or production, but can be used for any other type of environment as required.

***Screen Name:** Architecture*
***Screen Number:** 14*
Script:

The diagram illustrates a typical IBM Business Process Manager Standard configuration with all the tools available. The architecture centers around the Process Center, which contains the Process Server and the Performance Data Warehouse, and its associated database.

***Screen Name:** Architecture: Process Servers*
***Screen Number:** 15*
Script:

The Process Center pushes all process development in the Process Center to the different Process Servers in a “hub-and-spoke” deployment approach. Typically there is only one Process Center, and it pushes the assets out to the many Process Servers that are associated with the Process Center. Each Process Server has its own process server component and performance data warehouse component.

Screen Name: *Architecture: Process Center tools*
Screen Number: *16*
Script:

The Process Center has two tools that are unique to this environment: the Process Designer and Process Center Console. Authors create processes in the Process Designer, and the Process Center Console is used for process governance and administration.

Screen Name: *Architecture: Administrative consoles*
Screen Number: *17*
Script:

The process and business performance administrative console and the WebSphere Application Server administrative console are used in the Process Center to administer the server. The process portal allows process participants to create and work on the tasks that are assigned to them. In the Process Center, the Process Portal is used to debug process applications and playback the current version, or “tip,” of your process.

Screen Name: *Architecture: Process Server tools*
Screen Number: *18*
Script:

Process Servers, such as staging, test, production, and others, each contain their own set of the applications that are found in the Process Center. Each Process Server has a process and business performance administrative console, a WebSphere Application Server administrative console, and the Process Portal. These applications are dedicated to the server with which they are associated. A process participant uses the Process Server Process Portal to create instances or work on the process tasks. For example, testers use the test environment to create process instances and to see the tasks that are assigned to them. This Process Server is independent of the production environment, which has a different version of the process with production tasks that are running on the server.

Screen Name: *Unit 6: IBM Business Process Manager architecture and toolkits*
Screen Number: *19*
Script:
Topic 2: Installing process applications on a Process Server

Screen Name: *Installing process applications on a process server*

Screen Number: *20*
Script:

When the development is complete, copy those assets from the Process Center to a Process Server. This process is commonly called installing or deploying a snapshot to a Process Server.
.

Screen Name: *Installing process applications on a process server*
Screen Number: *22*
Script:

To deploy assets to a Process Server, developers must first create a snapshot of the process application. Snapshots record the state of library items within a process application or track at a specific point in time. You can create snapshots in the Process Center Console or in the Designer view. Snapshot management, such as installing, exporting, and archiving, is performed in the Process Center Console. More information about creating snapshots is covered in Unit 3.

After the snapshot is created, then it must be exported or deployed to the process server through the Process Center Console.

Installation services can be created to help the deployment of the snapshot. A governance process that reacts to the status change of a snapshot can also be created by using the System Governance toolkit.

When existing instances of a process exist on a process server, careful consideration must be made to the migration of those instances

To learn more about server configuration and administration, IBM Education offers classes on IBM Business Process Manager Administration and WebSphere administration.

Screen Name: *Unit 6: IBM Business Process Manager architecture and toolkits*
Screen Number: *21*
Script:

Topic 3: Understanding Toolkits. Process Designer authors use toolkits to share library items across process applications, and to assist developers in organizing their assets. The System Data toolkit is a standard feature of the installation, and it contains many helpful artifacts and standard data types that authors use during development.

Screen Name: *Understanding toolkits*
Screen Number: *22*
Script:

Toolkits are a collection of library items that can be used across numerous process applications. If you have a resource that can be reused, it is helpful to move it into a toolkit. When a toolkit is updated, all of its dependencies also show those updates. Toolkits can contain other toolkits as depicted in the picture with Toolkit 2 and Toolkit 4.

Screen Name: *Preinstalled toolkits*
Screen Number: *23*
Script:
IBM BPM comes with five toolkits preinstalled. Each provides unique capabilities to the developer.
The Content Management toolkit contains the assets that control the document attachments functionality that is used on coaches. These documents can be stored locally, or can be saved to an external content management system.
The System Governance toolkit contains data and services to govern the deployment of process applications to the other environments.
The Dashboards toolkit contains all the system dashboards that are used inside of IBM Business Process Manager.
All of these assets are easy to use and readily available to developers, but what happens when you want to change a stock dashboard? Developers can create tailored operational dashboards to offer enhanced visibility for process owners, team leaders, and business process users.
Because any upgrades of IBM BPM might include changes to the toolkit, if you require a customization to one of the assets in the toolkits, copy the artifact into another toolkit and modify it as needed. Therefore, any changes to the base toolkit do not affect your custom development.

Screen Name: *The System Data toolkit*
Screen Number: *24*
Script:

During installation, the System Data toolkit is installed. Each process application that is created is automatically linked to this toolkit. The System Data toolkit contains many services that assist developers to implement their processes, reducing the amount of custom development required.

Screen Name: *The Coaches toolkit*
Screen Number: *25*

Script:
Another special toolkit that assists you in developing your coaches is the Coaches toolkit. This toolkit contains all the standard components implementers use to build the web-based or mobile interfaces that are used to complete tasks within a process. The assets that are contained in the Coaches toolkit are integral to the rapid application development capabilities of the IBM Business Process Manager process designer.
You learn more about building coaches in playback 2.

Screen Name: *Creating and importing toolkits*
Screen Number: *26*
Script:

Toolkits can be created in the Process Center Console. They can also be created when you move items to a toolkit by using the IBM Process Designer. You can share toolkits internally, or they can be exported and shared with other IBM Business Process Manager installations. There are several community-created and support-created toolkits available on the community wiki. These toolkits can help you get started, but have different levels of support provided.

Screen Name: *Toolkit dependencies and snapshots*
Screen Number: *27*
Script:

To share a toolkit, you must create a snapshot of the toolkit. When it is created, other developers can then create dependencies on the toolkit. The toolkit can now evolve. When a new snapshot is taken, all of the process applications that are dependent on the toolkit receive a notification. Developers can then test their development so the new toolkit updates do not cause other errors.

Screen Name: *Demonstration - Understanding toolkits*
Screen Number: *28*
Script:

This instructor walkthrough covers:

- The toolkits tab
- How to create a toolkit
- How to take a snapshot of a toolkit
- How to import the coach bonus toolkit

- How to make a process application dependent on a toolkit snapshot

Click ***start demonstration*** to begin.

Screen Name: *Exercise 9 – Creating a toolkit*
Screen Number: *29*
Script:
Other process applications or other development projects can reuse many of the artifacts you create in the upcoming course. Create a toolkit to store and share these assets.
After completing this exercise, you should be able to:

- Create a toolkit
- Take a snapshot of a toolkit
- Import a toolkit
- Add a process app dependency to a toolkit

Screen Name: *Unit summary*
Screen Number: *30*
Script:

Having completed this unit, you should be able to:

- Describe IBM Business Process Manager product components
- Describe the relationship between the Process Center and the runtime environments or Process Servers
- Identify the process server and the performance data warehouse
- Create toolkits for easy reuse of assets

Unit 7: Conduct Playback 1

Screen Name: *Unit introduction*
Screen Number: *1*
Script:

Playback 1 focuses on enabling the model that is created during playback 0. After you implement gateways and associate swimlanes to participant groups, the process will flow down the right paths. Tasks are created and assigned to the appropriate users.

During the playback with the project stakeholders, it is common to receive suggestions for changing different aspects of your development. Business feedback is welcomed, and it is an intended byproduct of the playback. The point of incremental development and process validation is to catch problems or incorporate changes to the development. It must be done before it is too late and too costly to correct problems.
Playback 1 is unique because you might be forced to return to Playback 0 to make high-level process corrections, reimplement, and return to Playback 1. This process might happen more than one time before all the stakeholders can agree upon a high-level process. You will not encounter a return to a prior playback after the products from Playback 1 are solidified.

Always keep in mind the consequences when you consider changes to your process. Most changes affect the timeline and the scope of the project. It is always a judgment call to determine whether something is in-scope or out of scope. If a change is necessary, it is better to adopt those changes as early as possible during the development lifecycle.
If a change adds significant time to the overall project effort, a good practice is to maintain a change request log. Add these changes to the backlog for another release cycle. Road blocks can occur in the development lifecycle. It might be an indicator to determine whether that element must be adopted during this release, or sent to the backlog for future releases.

Screen Name: *Unit 7: Conducting playback 1*
Screen Number: *2*
Script:

After completing this unit, you should be able to:

- Describe the differences between Process Flow Data and Business Flow Data
- Add variables to a BPD
- Implement gateways to control process flow

- Describe participant groups and process lanes
- Implement routing for tasks
- Assign an expert group to an activity
- Expose a process app to a participant group
- Validate process flow

Screen Name: *Unit 7: Conducting playback 1*
Screen Number: *3*
Script:

This unit covers the following topics:

- Managing variables and data flow
- Implementing timer events
- Implementing gateways
- Routing tasks
- Validating process flow

The first topic covers managing variables and data flow.

Screen Name: *Two types of process data*
Screen Number: *4*
Script:

Two types of process data are used in Business Process Manager:

- Flow data
- Business data

Screen Name: *What is data flow?*
Screen Number: *5*
Script:

Flow data moves the process along. Data elements that use decision points on process and service diagrams are the most obvious examples of flow data. When a token is at a decision point, the value of each of the data elements is used to determine the next paths to take.

Flow data elements, however, go beyond the data that is needed to drive process decision points. Flow data includes all of the data that is used to determine:

- Which activities to complete
- Who completes each activity
- When an activity is due, or when an activity is escalated

Flow data must be identified early in the implementation process. Generally, by the end of playback 1, your flow data must be identified and implemented. Flow data gets the right activities to the right participants at the right time. Without flow data, the process cannot work.

Screen Name: *What is business data?*
Screen Number: *6*
Script:

The business data is more difficult to define than the flow data. In general, the business data provides the context of the activity to each process participant. The data is used to make it clear to each process participant what they are actually working on.

For example, a customer service representative knows, based on the activity that was assigned, that they are working on an insurance claim. The business data tells users the claim they are working on by claim type, claim number, customer description, and claim description.

Screen Name: *Declaring three kinds of variables*
Screen Number: *7*
Script:

Variables in a business process definition (BPD) or service can be declared in three different ways:

- The current BPD or Service requires the Private variables. The parent BPD or Service does not know or does not need Private variables. The value of a Private variable might still be of interest to any nested BPDs or nested Services, but the parent cannot access the private variables directly.
- Input variables are values that you can pass into the current BPD or Service.

- Output variables store values that are passed out from a BPD or Service to a parent BPD or Service.

The value of a private variable can still be of interest to any nested BPDs or nested services. Private variable data is not available to nested BPDs or services unless it is declared and mapped as an input or output. To identify which type of variable to declare, determine whether the current BPD or service needs the data. Identify whether the variable needs to be available after you leave the start event (input). Also, determine whether the variable needs to be sent out of the current BPD or service to a higher-level BPD or service (output). If the answer is neither, then it is a private variable.

Screen Name: *Standardizing Variable Names*
Screen Number: *8*
Script:

Create variable names that begin with a lowercase letter. This practice makes it easier to distinguish between a variable and its variable type, which begin with an uppercase letter.
Capitalize the first letter when you create a variable type, but use camel case for the instantiation of the variable. Standard variable types (Date, String, Integer) all follow this same naming convention.
If the variable name you choose consists of only one word, spell that word in all lowercase letters. If the variable name consists of more than one word, capitalize the first letter of each subsequent word. Remember, all variable names are case-sensitive.

Screen Name: *Demonstration - Declaring variables*
Screen Number: *9*
Script:

This instructor walkthrough covers understanding variable types and standardizing variable names.
Click ***start demonstration*** to begin.

Screen Name: *Understanding namespaces*
Screen Number: *10*
Script:

All variables in Business Process Manager are JavaScript objects. Business Process Manager uses namespaces to organize these objects, their functions, and their methods.

Examples of the available namespaces:

- tw: The top-level namespace
- tw.local: Access and update BPD-local and service-local variables
- tw.system: Access system features and functions
- tw.object: Used to initialize complex IBM Business Process Manager business objects

---

Screen Name: *Unit 7: Conducting playback 1*
Screen Number: *11*
Script:

Topic 2: Implementing the intermediate event: Timer. Now that timer events are modeled in playback 0, the control over the functions of the timer must be implemented. Next, implement the timer to fire based on the business requirements.

Screen Name: *Inline intermediate event: Timer*
Screen Number: *12*
Script:

There are two types of timer intermediate events: attached evens and sequence flow events. The sequence flow timer intermediate event is set to wait for a designated time before you continue down the sequence flow path.

The following event details are implemented to control the behavior of the sequence flow timer intermediate event:

- Trigger On: Specifies when the Timer Event starts.
- Custom Date: Use JavaScript to calculate and specify a date.
- Before/After Difference: Specifies the amount of wait time before the timer event fires.
- Tolerance Interval: Specifies an extra delay if work is in progress. This interval is measured one time.

The only way to repeat a sequence flow timer intermediate event is to have the process return down that flow line. In this case, the timer repeats as it did originally. The best use of a sequence flow intermediate timer event is to pause the process instance for a specified amount of time. The pause happens before the process can continue to the next flow object.

Screen Name: *Attached timer intermediate events*
Screen Number: *13*
Script:

The attached timer intermediate event has a unique set of properties that differ from the sequence flow timer. These properties exist because the task that it is attached to the timer intermediate event directly affects the event. Attached timer intermediate events are also called boundary timer events.

In addition to the properties found on the sequence flow timer intermediate event, two other options can be set. The options that are seen here are displayed only when the timer intermediate event is attached to an activity:

Boundary Event Details

- Interrupt Activity: Enabling this option allows the BPD to close the attached activity after the specified amount of time elapses
- Repeatable: Resets the timer to count again after the specified amount of time elapses

---

Screen Name: *Demonstration: Implementing intermediate event timer*
Screen Number: *14*
Script:

This instructor walkthrough covers implementing timer events and outlining the differences between inline and attached timer event details.
Click start demonstration to begin.

Screen Name: *Unit 7: Conducting playback 1*
Screen Number: *16*
Script:

Topic 3: Implementing gateways. Up until now, any gateways that are modeled in Playback 1 would be functional. The process would flow down a path. But to control the path that is taken, developers must implement the gateway decision logic.

Screen Name: *Implementing a gateway*
Screen Number: *16*
Script:

Any gateways that are modeled in playback 0 are functional, as processes are able to flow down a path. But to control the path that is taken, developers must implement the decision logic that the gateway uses. In the playback process, it is necessary to demonstrate each path that can be taken. The logic on how that decision is reached is implemented in a later playback. When you specify the implementation for a gateway, you define conditions that control whether a path is followed during the running process. After you create the rules and integrate the data, you must test the newly functioning gateways by using the inspector view of the Process Designer.

The suggested practice for playback 1 is to avoid implementing Boolean (true or false) variables to implement decision gateways. This practice helps you to avoid problems with the gateway when more flows are added later. Use simple variables, such as String, or Integer, to drive all decision gateways. Developers can set the default value of the BPD variables and show the different paths that are being taken during playback 1. Document the gateway logic to help troubleshooting the process.

Screen Name: *Demonstration – Implementing gateways*
Screen Number: *17*
Script:

This instructor walkthrough is an overview of gateway properties and how to implement them in a business process definition.
Click start demonstration to begin.

Screen Name: *Unit 7: Conducting playback 1*
Screen Number: *18*
Script:

Topic 4: Routing tasks. The final task in playback 1 completes the goal of getting the right tasks to the right people at the right time. Variables are built, and those variables are used to drive decision gateways. When a task is created for a process participant to complete an activity, the task must be assigned to the right individual to complete the work.

You begin with an introduction to participant groups and then move on to examine process routing.

The first topic is participants and participant groups.

Screen Name: *Creating a team*
Screen Number: *20*
Script:

Teams represent the groups of users in your enterprise that can be assigned a task in a swimlane or assigned directly to an activity. The Process Designer can be used to assign members to a team. But, there might be other requirements to assign members in the different Process Server environments. System administrators can map the teams to security groups with the administrative screens on the various environments.

To learn more about server configuration and administration, IBM Education offers classes on IBM Business Process Manager Administration and WebSphere administration.

Screen Name: *Team and team members*
Screen Number: *20*
Script:

A team contains the users who complete the runtime activities that are modeled in each lane. Team lane assignments ensure that any activities that are not routed to a specific user have an automatic default assignment.

When you create a lane, each lane is assigned a default team that is called All Users. This default team contains all of the users of IBM Business Process Manager to allow for testing of your processes. A system lane is a lane which activities do not have a coach. The system runs the activity automatically. If a lane is supposed to be a system lane, select the Is System Lane check box. Afterward, add the System team in the Behavior section. System lanes are shaded a different color so they can be easily recognized.

Screen Name: *Routing Activities*
Screen Number: *21*
Script:

In many cases, you might not want an activity to go to the default participant group or you might need a more dynamic solution. For any activity with a service

(task) implementation, you can designate the users who must receive the runtime task. Use the Assignments option in the Property tab of the Activity.

Use the options available in Routing to designate who to route the Activity to (Assign To) and how it must be distributed (User Distribution). By default, the "Assign To" setting is set to Lane Participant. Tasks are assigned to the participant group designated for that lane. The two most commonly used lane assignment selections are: Lane Participant and Last User in Lane.
Last User in Lane can be used for the first activity in a top-level BPD. The runtime task is routed to the user who started the BPD in this case.

By default, User Distribution is set to None, meaning that tasks are not assigned to individual users. The task is assigned to the pool of potential users, allowing individual users to acquire tasks that are assigned to the pool.

This course focuses on only the most common Routing options used.

Screen Name: *Team filter service*
Screen Number: *22*
Script:
There are times when a developer does not want the whole team to be assigned to a task, but rather a subset of the team. There also might be situations where a team must be created or filtered at run time. The team might be filtered based on business data or process data. Use a team filter service to create a subset of a team to assign a task to. The input of the service would be the entire team, and the output would be the filtered list. You can also add more input parameters as needed to determine the new team. The filtered team subset is returned as a Team object. This object type is defined in the system toolkit.

Screen Name: *Team retrieval and filter service templates*
Screen Number: *23*
Script:
The team retrieval and filter services are a type of integration service. This service returns a list of users to create an ad hoc team at run time. The filtered team subset is returned as a Team object. This object type is defined in the system toolkit.
The easiest way to create a team filter is to start from a team filter service template included in the system data toolkit. This template outlines the basic inputs and outputs of the service.

Screen Name: *Identifying expert users for an activity*
Screen Number: *24*

Script:

Business users can collaborate or request assistance from a set of expert users that are associated with a particular task or activity. This list of experts is displayed in the Experts panel in the Process Portal environment.
Only activities that contain a human service can be assigned experts.
The set of experts that is listed for an activity is defined:

- Based on historical analysis, the users that complete the activity in the past
- Users that belong to a participant group that is explicitly specified as an expert group for this activity.

You can assign an experts group on the Assignments tab in the Properties view. Your BPM administrator can configure the participant groups at run time. It ensures that the correct users are identified as experts for the activity in the Process Portal.

Screen Name: *Demonstration: Teams and routing*
Screen Number: *25*
Script:

This instructor walkthrough is an overview of users, teams, routing tasks, and assigning an expert group.

Click start demonstration to begin.

Screen Name: *Exercise 10 – Creating playback 1 assets*
Screen Number: *26*
Script:

This exercise is the step-by-step process for creating playback 1 assets.
After completing this exercise, you should be able to:

- Create simple variables in a BPD
- Implement timer intermediate events on a process
- Implement gateways for a process
- Implement routing for an activity

Screen Name: *Unit 7: Conducting playback 1*
Screen Number: *27*
Script:

Topic 5: Validating process flow. The final section for this unit is validating the process flow.

The process application is ready for playback 1 validation. This session is the opportunity to demonstrate everything that was created thus far.

- Process flow variables are driving decision gateways
- Tasks are created and assigned to the right participant groups
- The task routing process is correct
- The process follows the correct path

It is important to gather all the process stakeholders to validate the process that was created. Verify that it meets the business needs outlined in playback 0. Verify that the process works as modeled and as expected. Set the process flow variables and watch the process follow the correct path on your decision gateways. If it is necessary to change the high-level process, then move back and complete playback 1. After you receive confirmation that the model meets the goals of the current release, the process application is ready for playback 2.

Screen Name: *Exposing a process to a team*
Screen Number: *28*
Script:

Before you demonstrate the process, create an instance of your process. Expose the process application to a team so that the group participants can create a process instance. Go to the **Overview** tab of the BPD. Select a team that has access to start this BPD, and assign it to the **Expose to start** option. The quickest way to demonstrate the process is to expose the BPD to the **All Users** team. After the BPD is exposed to all users, any user with a valid account can start the process.

Screen Name: *Playback 1*
Screen Number: *29*
Script:

To demonstrate the process follows the correct flow out of the gateways, set the value of the process flow variables. Use the **Has Default** option to set the value; using this option demonstrates that the process follows a designated path. Gather all the stakeholders in a room and walk through the process.

Demonstrate the process flows down the different paths and validate the tasks that are assigned during those paths are correct. Use the portal inbox to show tasks that are assigned to the right users. Complete those tasks so stakeholders see how users interact with the system.

Screen Name: *Exercise 11 – Conducting playback 1*
Screen Number: *30*
Script:

This exercise comprises the step-by-step process of conducting the playback 1 of a BPD.
After completing this exercise, you should be able to:

- Lot on to the Process Portal
- Create an instance of a process
- Demonstrate that the process follows the different paths modeled

---

Screen Name: *Unit summary*
Screen Number: *31*
Script:

Having completed this unit, you should be able to:

- Describe the differences between process flow data and business flow data
- Add variables to a BPD
- Implement gateways to control process flow
- Describe teams and process lanes
- Implement routing for tasks
- Assign an expert group to an activity
- Expose a process application to a team
- Validate process flow

Unit 8: Conduct playback 2

Screen Name: *Unit introduction*
Screen Number: *1*
Script:

This unit covers playback 2. Playback 2 builds upon the foundation that is established in playbacks 0 and 1. The next piece of the puzzle is to concentrate on business data. It is the data that users view and input into web pages when they complete their tasks. These screens are called “coaches” in Business Process Manager. Authors can use the tool to create a usable front end in a short amount of time. Playback 2 does not focus on how the front end looks, or on any usability enhancements; these aspects are considered later, in playback 4. The goal is to have business data that flows through the different screens that are presented, and that flows between the activities in the process. It is important to ensure that the process captures all of the data that users need to complete their tasks. The overall process must be able to complete as well.
Always plan for future playbacks, but concentrate your development effort on the current playback. If authors have an integration point or a menu that is presented on a coach, you can focus on planning how to get the data. The implementation of the coach fields that are used to access the data is completed later. Focusing on the task at hand makes the development more agile. It helps avoid rework if artifacts need to be changed because they were created too early. By following the playback methodology, the amount of rework is reduced. Realistically, however, rework is never eliminated no matter which development methodology is used.

Screen Name: *Unit objectives*
Screen Number: *2*
Script:

After completing this unit, you should be able to:

- Build a business object
- Initialize a complex object and a list
- Build a service
- Define and implement guided user interactions by using coaches
- Implement a service for an activity in a business process definition (BPD)
- Map variables between a nested service and an activity in the overlying BPD
- Use object methods

- Create a snapshot of a process app for deployment

Screen Name: *Unit 8: Conducting playback 2*
Screen Number: *3*
Script:

This unit covers the following topics:

- Creating a data model
- Building services
- Building coaches
- Implementing services in a BPD
- Creating a snapshot for deployment

Screen Name: *Defining the business object data model*
Screen Number: *4*
Script:

Business objects, or variables, are used to represent the business data that is relevant in the context of your business process. Create a data model that accurately reflects the business data and its structure. Do not allow external elements influence or affect the development of your process business object variables that belong in your data model. The following list must not affect your variable creation:

- Existing web pages
- Other systems
- Coaches
- Web services
- Database tables

It is best to loosely couple your business objects from the external sources. Use either an Enterprise Service Bus or an internal service to map variables to other variable schema. That way any changes to external variable structures limit the affect your data model, and vice versa. Changes can be made in the adapter/translator, limiting the changes necessary for your library artifacts.

Screen Name: *Building a complex business object*
Screen Number: *5*
Script:

When the system data toolkit business objects cannot meet your requirements, you can create custom variable types. You can create a custom business object type by using a base business object type or by defining a new complex structure.
You can create rules about complex data that is nested, or hierarchical. Data that is referenced within the text of a rule is not limited to simple object types such as string, integer, or date. You can create complicated rules with nested object structure. Click **Add** and create the complex object that is made from simple or other complex object types.

Screen Name: *Building the data model*
Screen Number: *6*
Script:

After you define the data, organize your data into different logical units. An example might be a street address, city, and state that becomes a unit called addressBlock. Some organization is obvious, but other data is more difficult to organize. Look to existing data models in your company to help you create the object.
An example of a business object is a structure that contains multiple elements all pertaining to the same “subject”. It is modeled after what the structure represents (for example, a name and phone number would be relevant to an employee). There is a mixture of art and science to create good complex objects, but keep in mind simplicity and reuse when you create these objects. Do not bog down your complex objects with elements that might never be used, even if other objects in the organization might include those elements.

Screen Name: *Standardizing the business object naming convention*
Screen Number: *7*
Script:

Business object type names begin with an uppercase letter. Adopt this convention to differentiate business objects and business object types. If the business object type name consists of more than one word, capitalize the first letter of each word. In this example, the employeeData variable uses the business object type EmployeeInformation. Business objects and object type names are always case-sensitive.

Screen Name: *Refactoring*
Screen Number: *8*
Script:
Over time, applications change and business objects, their attributes, and variables might be renamed. However, many parts of a business process might reference or have a dependency on a business object, an attribute, or a variable. Renaming any object, therefore, can produce unexpected results.
For instance, suppose you created an object type address that contains city, state, and postal code elements. Later on, your company goes international and wants to specify the current object as a US address and add other addresses that belong to other countries. If you change the object name, anything that references the original address object throws an error, and you would have to manually parse through all the objects in your process application to correct the errors.
Refactoring offers a one-step process to update references to the changed variable name. When you use the combination Alt + Shift + R after renaming an object, a rename window will open and guide you through targeting all the things in the library that use the old object's name.
Refactoring applies to:

- Renaming objects in processes and services
- Renaming attributes in processes and services
- Renaming local variables in a process and its related services

Refactoring is available to those objects referenced under the Variables tabs in the BPDs and services in your Process Application. Unfortunately, one limitation is that any JavaScript code found in coaches are not updated.

Screen Name: *Complex objects and lists*
Screen Number: *9*
Script:

You can declare any business object to be a list, or an array, of a business object type. Instead of containing only one of the types declared (String, Date, Integer), your business object holds many of the same business object type. Therefore, if you wanted to create a business object that contained multiple Integers, you might create a Private list business object of type "Integer".

To create a business object list, check the check box next to "Is List:", and the list object will add the "(List)" designator after the business object type in the variables that are listed on the left.

After you create the list variable, you must initialize the variable before you can use it by using. listOf.

Screen Name: *Initializing complex objects and lists*
Screen Number: *10*
Script:

In IBM Business Process Manager, all complex business objects and all lists (arrays) must be initialized before you use them in a BPD or service. Before you use a complex business object, initialize it by inserting the JavaScript that is shown into a server script object. Server scripts are explained later in this unit.

As you can see, the name of the complex business object that is being initialized is tw.local.requisition. The name of the complex business object type (variable type) is Requisition.

Your business object type (variable type) might include elements that are themselves business object types other than the simple types (sting, date, integer). You must initialize those subobjects as well before you use them.

You must also initialize lists before you use them by using .listOf. This approach works for simple and complex business object types. If you have a list of Strings, you can initialize them using a script as shown by the example.

When your list is created, then you must add elements to the list. Assign the object directly to the list with the index, as shown with the first example by assigning yourStringList to the first element of the list. You can also add objects to the end of the list by using the construct that is shown in the second example by using .listLength as the index. Therefore, if there are no elements added to the list, the list was recently created and empty. The system adds the first element to the list in position zero, or the first element of the list. Both examples accomplish the same task, but the second one can be used in a loop to add elements to the end of the list.

Screen Name: *Nested processes and variables*
Screen Number: *11*
Script:

Developers must be careful to consider how variables are passed between the parent and the nested process when you use nested processes. Subprocesses, event subprocesses, and linked processes handle variables differently. A subprocess has access to the data of the parent process, and data mapping is not required to pass data into or out of the subprocess. You can also declare private variables within the subprocess that are not visible to the parent BPD.

Event subprocesses are just a special type of subprocess, and variables that are defined in the parent BPD are accessible by the subprocess. An event that

occurs in the parent process triggers an event subprocess. It is not just a step in the process as part of the process flow.
Linked processes require developers to create input and output variables in the nested process and to map variables from the parent process to the linked process. The variables must be of the same variable type when mapping.

If you are debugging a process and find variables that are not available in a BPD or service, check whether the variables must be explicitly passed. If so, check whether the variables are mapped correctly.

Screen Name: *Sharing a complex business object*
Screen Number: *12*
Script:

When you create a custom business object in a process application, that object is available for all BPDs and services included in the process application. If you want to share a custom business object across process applications, create or store the custom object in a toolkit. Then, create a dependency on that toolkit from the process applications that require the variable.

Use the Shared Object check box if the business object and its values must be accessible to other instances at run time. It designates that the data must be shared. The business object becomes a shared object. Shared business objects apply only to a complex structure type. The data within a shared business object is shared between business processes and tasks.

The data within a shared object is persisted to the database when:

- The shared object is created
- The business process or task is persisted to the database
- When the JavaScript method save() is done on the shared business object

---

Screen Name: *Demonstration - Creating a data model*
Screen Number: *13*
Script:

This instructor walkthrough covers:

- Building a complex business object
- Building the data model
- Standardizing the business object naming convention
- Declaring lists

- Initializing complex objects and lists

Click start demonstration to begin.

Screen Name: Unit 8: Conducting playback 2
Screen Number: *14*
Script:

Topic 2: Building services. After you model the business process, it is time to start thinking about these processes as sets of interactions. The goal of this playback involves building interactions. These interactions represent a critical piece of BPM. Do not solely focus on the chains of activities with their sets of inputs and outputs. Authors also need to think about the protocols and agreements that are made between the participants.
Services provide a methodology for choreographing these protocols and agreements. Choreography is a more abstract notion of process. It is used to describe the “interactions” of collaborating entities, each of which can have their own internal orchestration (modeling) processes. Next, build services in a business process definition.

Screen Name: *What are services?*
Screen Number: *15*
Script:

Services are made up of steps that define what happens when service (task) activities are triggered in the process. Each step enables the service to do a different job, from integrating with an external data source to generating forms, or coaches.

Screen Name: *Creating services*
Screen Number: *16*
Script:

The IBM Process Designer is where authors build reusable services that implement the activities in a business process definition. When you use the Process Designer, services differ from BPDs in that there are no swimlanes, but the interface is similar to creating BPDs. The service palette on the right is customized with what items you can use to build the service. For example, in Playback 2, you examine human services. A coach can be added only to a human service. Each service type is designed to accomplish a specific type of task. If you need help in determining which service you must create, consult appendix C in the student manual for a detailed service chart.

Screen Name: *Demonstration - Building services*
Screen Number: *17*
Script:

This instructor walkthrough covers:

- The different types of services available in IBM Business Process Manager
- Using process data in services

Click start demonstration to begin.

Screen Name: *Unit 8: Conducting Playback 2*
Screen Number: *18*
Script:

Topic 3: Building coaches. A human service presents a form for a process participant to interact with. This form in IBM Business Process Manager is a web page that is called a “coach”. This unit analyzes the fundamentals of building coaches, and enables developers to create coaches in a short amount of time.

During Playback 2, developers must not look to alter the visual presentation or add any extra functions to their coaches. Developers must concentrate on creating forms that can be filled out, and ensuring the data is posted back to the server. Make sure that the process gathers all the necessary data from the participants in the process to complete the overall process.

Screen Name: *Coach overview*
Screen Number: *19*
Script:
There are two types of user interface in IBM Business Process Manager: a dashboard to provide a heads-up view of all the accomplished work and those waiting tasks to be accomplished, and the pages that are used to complete tasks. How can users input the data necessary to accomplish their task inside of a process? Coaches, or a standard web page brought up in the user’s browser, are used to gather the necessary data for users to complete their tasks. Developers build coaches that provide a wizard-like approach for process participants to complete their tasks.
When an activity on a BPD is assigned to a team or a user, a coach is rendered in the browser to allow the user to provide the process the necessary data to complete the task.
Building a coach is similar to building BPDs and services, and uses a “what you see is what you get”, or WYSIWIG interface. When the coach is then rendered

for the user, the development view is similar to what the user uses. Therefore, developers do not need to be HTML experts to create basic forms.

Screen Name: *The coach designer interface*
Screen Number: *20*
Script:

The coach designer is the fastest way to build the interface for users to complete their tasks inside a process. Developers build their coaches objects that are called coach views. Similar to web pages, coaches use divs and spans to control the layout of the webpage. Coach views and coaches contain the same HTML, JavaScript, and CSS of a standard webpage. The coach designer enables developers to build the interfaces without having to hand-code all the objects, tags, and server-side plumbing necessary to move data in and out of the client-side webpage. The coach designer allows developers to concentrate on building pages and reduces the burdens of coding session management and security elements on the server and client.

Screen Name: *Sections*
Screen Number: *21*
Script:

Sections control the layout of your controls after they are rendered in the browser, and are made of HTML div and span sections. Drag the horizontal and vertical sections onto the coach palette, and then add the controls to the sections. A tabs section can also be used to provide a way to group similar sections. All the sections can be nested within each other.

Section properties are separated into three general categories: General, Configuration, and HTML Attributes.

The general property describes the general layout of the section. The configuration controls the look of the section. The default options can be set to show or hide the border around the section, and to square or round the corners of the section. The HTML section allows developers to add classes and attributes to the HTML objects rendered in the browser.

Screen Name: *Controls*
Screen Number: *22*
Script:

Controls are objects that can be placed on the palette of the coach. Every control is made of a coach view. Some examples of controls are a button, check box, input box, date time picker, select control, and other standard HTML objects. The mobile-ready set of objects are formatted for viewing on a mobile platform. Although the standard objects cover most of your needs, IBM Business Process Manager allows developers to customize their own controls. Any control or custom code can be added manually by using a Custom HTML block on the coach. The blocks can include JavaScript, CSS, or custom HTML elements. When the controls are laid out, it is easy to copy and paste them into a coach view. Then, the view can be shared with other coaches across your Process App or other installations.

Screen Name: *Coach view object property settings*
Screen Number: *23*
Script:
Coach view objects, whether sections or controls, contain property settings for use in the coach. The property settings are separated into four categories:

- **General**: describes the common properties of the object, such as the control ID, and the behavior, such as binding and label visibility
- **Configuration**: allows developers to modify the configuration attributes of an object
- **Visibility**: sets the visibility of the object through a variable value, a rule, or a script
- **HTML attributes**: allows developers to add classes and attributes to the HTML objects rendered in the browser

---

Screen Name: *Adding controls from server-side variables*
Screen Number: *24*
Script:

There is a quick and easy way to add controls to a coach that are automatically bound to server-side variables. The first step is to make sure that you have your simple or complex variables (Input, Private, or Output) defined for your service. When you create a coach, you see the variables that are defined for your service are listed on the right side. Click and drag your variables onto the coach palette, and controls are created and bound automatically to the variables and any subobjects if you are moving a complex object.
There might be controls on the coach that control the process flow. Add the process flow variables necessary to control the process directly on your coach. It adds an input box that is bound to the process flow variable when you run your coach. During the process playback, you can enter the process flow data into that input box to control your gateways.

Looking forward to Playback 3, consider whether you plan to implement a rule service or some other logic to drive process flow. The rule service might be based on the business data that is obtained on this coach. Add both the process flow variable that is used to control the gateway during this playback for troubleshooting purposes. Input the expected results of your logic in the input box of your coach during the process playback.
Later on in Playback 3, you must implement the required logic to decide on business data and create the process data required.
When you “beautify” your coach in Playback 4, remove the process flow variable input on your coach before your coaches are finalized.

Screen Name: *Buttons and sequence flow*
Screen Number: *25*
Script:

Next, you need the ability to submit the coach to the server and move the service along. Buttons allow the form to post the data back to the server from the client. Developers can model other abilities such as saving progress or retrieving external data.
After you add the wanted buttons to your coach, sequence flow must be connected in the service Diagram. Each button requires a new line to be drawn in the diagram. Each button item corresponds to one line or flow if the button is configured to have a flow.
In the slide, a line is sequence flowed from the coach to the End event. There is a button on the coach that is called **Next**, and another called **Save Progress** going to a modify task service.

Screen Name: *Demonstration - Building coaches*
Screen Number: *26*
Script:

This instructor walkthrough provides an overview of:

- The coach interface
- Investigates the anatomy of a coach
- Demonstrates working with coach sections and controls
- Shows how to add controls and buttons
- Steps through adding buttons and sequence flow.

Click start demonstration to begin.

Screen Name: *Unit 8: Conducting playback 2*
Screen Number: *27*
Script:

Topic 4: Implementing services in a BPD. Now that coaches are built, authors can enable BPD-level activities to use human services. In this unit, activities are implemented by using the artifacts that are built.

Screen Name: *Implementing services*
Screen Number: *28*
Script:

When you first add an activity to a lane in a process, notice the activity implementation. The system automatically implements the activity with a default human service or a default system service. To specify what happens when an activity is triggered at run time, you must attach a service by using the implementation menu in the Properties view.
Attach any of the services you create to the activity on the BPD. When a token flows to this activity, the system creates a task, and the user completes the task. The system might also complete the designated service.

Screen Name: *Data mapping*
Screen Number: *29*
Script:

Data mapping is used to pass the values of variables between an activity (task) in a BPD and a service. When you attach a service to an activity, the Data Mapping section needs to be populated with the input and output variables of that service. The Data Mapping section is used to map the input and output variables of the service to variables in your process. Runtime variable values can be passed to and from the service with these settings. Variables can either be automatically mapped to your process variables by pressing the auto-map button, or you can manually specify the process variables to be mapped.

Screen Name: *Understanding object methods*
Screen Number: *30*
Script:

All objects inside of IBM Business Process Manager have methods that are associated with them. If you are familiar with object-oriented programming, then you are familiar with using methods on the objects. To access the object methods, you must always first use the object namespace followed by the object, and finally the method you would like to access.

For example, the JavaScript method toLowerCase might be called on an object of type String. In this example, the results of the call are used to convert the value of tw.local.myString to all lowercase. Certain times the method affects the object directly, and other times the method returns an object that you must map to another object. Use the auto-complete feature when you determine how an object method is used, as the system displays how to use the object method.

Screen Name: *Demonstration - Implementing services*
Screen Number: *31*
Script:

This instructor walkthrough demonstrates how to implement a service and data mapping in Business Process Manager. It also demonstrates running and debugging processes and services with the Process Designer.
Click start demonstration to begin.

Screen Name: *Exercise 12 – Creating and implementing a coach service*
Screen Number: *32*
Script:

This exercise is the step-by-step process for implementing the created services to the appropriate activity in a BPD.

After completing this exercise, you should be able to:

- Determine and organize data when provided with a written process
- Add business objects and object types
- Create a new human service
- Add variable and variable types to the service
- Create and configure a coach to obtain process participant input
- Add coach controls to control process flow
- Create a human service and coach for the General Manager review activity
- Implement an activity by attaching a service and mapping data

---

Screen Name: *Unit 8: Conducting playback 2*
Screen Number: *33*

Script:

Topic 5: Creating a snapshot for deployment. At major milestones during a playback schedule, it is necessary to create a snapshot of the process application. In a typical engagement with a testing team, the end of playback 2 would be an opportune time to deploy a snapshot to the test environment. You can incrementally test and validate the created artifacts when you deploy snapshots. Snapshots can also help you identify any bugs in the development process to date.
To deploy a process application, you must create a snapshot of the process application. That snapshot is the artifact that is deployed to a runtime environment by an administrator.

Screen Name: *Creating snapshots: IBM Process Designer*
Screen Number: *34*
Script:

You can capture and record a process application at a specific point in time when you use snapshots. Snapshots are often taken to use during playbacks or at other milestones in development.
You can compare different snapshots and revert to previous snapshots as needed during development.
For example, a developer can fix a problem with a service and take a snapshot at that point. Then, a different developer can make several more changes to the same service and take a new snapshot. The project manager might compare the two snapshots to determine which changes were made when and by whom. If the project manager decided that the additional changes to the service are not worthwhile, they can revert to the snapshot of the original fix.

Snapshots are also used for installing or moving applications from the Process Center to other Process Servers, such as testing or production.

A snapshot can be taken in IBM Process Designer with the button in the upper right corner. All prior snapshots for a process application are listed in the lower left corner window.

Screen Name: *Creating snapshots: Process Center*
Screen Number: *35*
Script:

Snapshots can also be taken in the Process Center console. When the permissions are set, administrators can also create snapshots of a process application for deployment to a process server with the process center console.

Screen Name: *Guidelines for using snapshots*
Screen Number: *36*
Script:

The following are some guidelines for using snapshots:

- Snapshots can use a large amount of space in a database. Agree on snapshot intervals in the organization, and take snapshots at the agreed-upon milestones.
- Define a meaningful naming convention for snapshots, and use that naming convention for all projects.
- Work closely with Business Process Manager administrators to define a snapshot deployment and activation plan.

Screen Name: *Demonstration: Create a snapshot*
Screen Number: *37*
Script:

This instructor walk-through provides a deeper understanding of snapshots and demonstrates creating snapshots.
Click start demonstration to begin.

Screen Name: *Exercise*
Screen Number: *39*
Script:

This exercise is the step-by-step process of creating a snapshot of the process application.
After completing this exercise, you should be able to:

- Create a snapshot of your process application

Screen Name: *Unit summary*
Screen Number: *40*
Script:
You are now ready for Playback 2. The goal is to demonstrate flowing data from one coach to another and from one task to another inside of the BPD. All data

must be bound on the coaches. If the same data is shared across multiple tasks, playback participants see the moving data through the different activities.
Log in to the Portal and demonstrate the coaches. The screens have fields that require business data. If there are any questions around presentation, visibility, JavaScript functions, or other “enhancement” questions, record the questions to a log. During this playback, focus the discussion not on the presentation, but rather the data itself.

Presentation aspects for coaches can eat up a significant amount of development time. Therefore, it is important to spend this time to make sure the coach human works as designed, and return to the presentation concerns in Playback 4. That way, if timelines shrink or development falls behind, the emphasis lies in the correct behavior. If it becomes critical to deploy a version without the right presentation, a functional snapshot can be taken and deployed. The presentation enhancements can continue to be refined, and usability testing can run in parallel.

Having completed this unit, you should be able to:

- Build a business object
- Initialize a complex object and a list
- Build a service
- Define and implement guided user interactions by using coaches
- Implement a service for an activity in a business process definition (BPD)
- Map variables between a nested service and an activity in the overlying BPD
- Use object methods
- Create a snapshot of a process application for deployment

Unit 9: Playback 3

Screen Name: *Conduct playback 3*
Screen Number: *1*
Script:

Because of the work that was done previously, playback 3 now focuses on “real data, in real time.” This playback encompasses all of the integrations that are needed to drive the process. Up until now, any system lane activities were either not implemented, or they used mock services with fabricated data. Any data that needs to be looked up from a database or external source is now implemented and demonstrated. This data includes all incoming and outgoing integrations.

Looking ahead, make sure that you also include any lookup services necessary to drive coach functions for Playback 4. Playback 3 is the place to create any services that retrieve or send data to any Ajax services or select menus on the coach. Enabling those services on the coach comes in the next playback.

It is common to plan for integrations early on in the playback process. When you finally arrive to Playback 3, those integrations might not be available. By Playback 3, it is good to create all artifacts in anticipation of including the integration in your application. When the end point is created, swap out your mock services with the actual integration. That way your development can continue while you wait for your integration end points.

Screen Name: *Unit 9: Conducting playback 3*
Screen Number: *2*
Script:

After completing this unit, you should be able to:

- Create a decision service
- Create a message start event
- Create an enabling service
- Create and configure an undercover agent (UCA)
- Start a BPD with a message start event
- Organize assets with favorites, tagging, and smart folders
- Define the basic function of an integration service
- Identify the components of the IBM Business Process Manager integration architecture

- Describe how integration components interact with services
- Configure and define integration services for outbound integration
- Describe the differences between an environment variable and an exposed process variable

---

Screen Name: *Unit 9: Conducting playback 3*
Screen Number: *3*
Script:
This unit consists of the following topics:

- Creating a decision service
- Implementing message events
- Applying asset tagging
- Accessing and manipulating external data
- Exposed process variables and environment variables

Topic 1: Creating a decision service. Developers build a decision service when a decision or condition in a business rule is needed to determine which process implementation is started. For example, when a certain condition evaluates to true, Process Designer implements the associated activity or action.
Business analysts and business users who are rule designers rather than programmers might author the business-rules with the support of Process Designer. Business rule designers can express business logic by using rule syntax that resembles natural human language. This rule syntax is called Business Action Language (BAL), which is a declarative language that relates business concepts to business data and actions. A rule execution engine interprets business rules, which are an expression of business policy in a form that is understandable to business users. Business rules formalize a business policy into a series of “if-then” statements.

Screen Name: *Building a decision service(1 of 2)*
Screen Number: *4*
Script:

Business rules are included in a BPD by adding a decision service to the process. Add a decision service to a process application when the actions that must take place in your process depend upon one or more conditions.
When you build a decision service, follow these guidelines:

- Build your rule hierarchy so that rule conditions are ordered from most complex to least complex.
- Create a final condition that is a catch-all rule. This rule makes sure that the variable is assigned a value even if the prior rules do not apply.
- Consider encapsulating your rules in a single-function decision service. This encapsulation allows the service to be available to any other part of the process application that needs the same rule logic.

Screen Name: *Building a decision service (2 of 2)*
Screen Number: *5*
Script:

When you create a decision service, the options on the right side changes to show options that are unique to this service type. There are three types of components available to be used in a decision service. The BAL Rule or Business Action Language (BAL) rule, is a natural language technology. JRules Decision Service, which integrates with an IBM WebSphere ILOG JRules Rule Execution Server. The final option is the Decision Table, which contains a rule table. Each row in the rule table represents a Boolean condition that evaluates to true or false at run time. When a rule evaluates to true, the JavaScript expression that you provide as the rule action is run.

Screen Name: *Adding a BAL rule*
Screen Number: *6*
Script:

The Business Action Language (BAL) rule component provides a rule editor that allows rule designers to author business rules that use natural language technology. Using natural language, instead of JavaScript, to author rules means that no programming expertise is required to create business rules. The rules are easier for people to read and understand.
Add rules by using the plus sign at the top of the screen, and remove rules with the X next to the condition to delete. Move the rules up and down with the arrows to the right of the corresponding rule.

Screen Name: *Building a BAL rule*
Screen Number: *7*
Script:

You can use the BAL rule editor to build rules, add rule parts, statements, and fragments, and replace placeholders with variables and values. Use the

completion menu in the editor to insert or edit constants, values, parts, or fragments of rule statements. While you are creating or editing rules, the editor highlights errors to help you identify and resolve problems in your rules.

A business rule consists of some or all of the following parts in the following order: definitions part (optional), if part, then part, and finally the else part (optional).

When you create rules, the Content Assist box be your best friend. It provides you options to choose from to build your rules. If you ever get stuck, press Ctrl + Spacebar to view the menu and read through the suggestions and the descriptions of the options available.

Screen Name: *Demonstration - Create a decision service*
Screen Number: *8*
Script:

This instructor walk-through demonstrates building a decision service and adding a BAL rule.
Click start demonstration to begin.

Screen Name: *Unit 9: Conducting playback 3*
Screen Number: *10*
Script:

Topic 2: Implementing message events. Events can occur at all points of a business process and most likely affects its flow. Thus, a major part of implementing business processes is to know how to best handle and react to dynamic events. These events might be internal or external to the process. IBM Business Process Manager provides two components, message events and undercover agents (UCAs) to model and implement these dynamic interactions at run time. You now have an interface to accept incoming messages (undercover agents) and a listener (message event). That incoming message triggers the event and accomplishes a task or set of tasks.
In Business Process Model and Notation (BPMN), a message generally signifies any signal from outside the process. Like all intermediate events, the message intermediate event can be used in sequence flow or attached to an Activity.

Screen Name: *Message event introduction*
Screen Number: *10*
Script:

You implemented a timer event on a BPD, but what happens if a certain process requires instances to be created on a regular time interval? Because a timer event cannot create instances (it is an intermediate event, not a start event), you must look to another solution. Use a start message event that is enabled by a time-elapsed undercover agent (UCA).
In this example, a participant (a manager) is required to submit a monthly claim for their group expenses. The system generates a generic claim shell for the manager to complete, and the manager must complete the claim before it is submitted. You cannot use the intermediate timer event to create instances, so you must turn to the time-elapsed undercover agent.

Screen Name: *Message start events*
Screen Number: *11*
Script:

The message start event specifies that an incoming message starts a process at run time. When a message start event receives a message, a new instance of the business process definition is created. A unique BPD instance ID is assigned to it. The start message event has the exact same function as a regular start event. But, instead of a user whom creates the process instance through the portal, the message event creates the instance.

Screen Name: *Creating an enabling service*
Screen Number: *12*
Script:

To implement a start message event, the first step is to create an enabling service. A generic system service can be created with an “ES” suffix added to the name to distinguish it from other services. If variables are required for the start of the process, this service must create the output variables.
The output of the enabling service defines what data is sent as part of the message payload. Outputs of the enabling service are optional. There also cannot be a coach as part of the enabling service.
For this example, you created the variable orderDetails of type OrderDetails, and created the data with a server script inside the enabling service. When the enabling service is created, you use it to create the undercover agent.

Screen Name: *Creating an undercover agent (UCA)*
Screen Number: *13*
Script:

The next step would be to create the UCA. There are two settings for the schedule type: time-elapsed and on demand. Time-elapsed is similar to a cron job, in that it fires the UCA on a regular time schedule. On demand is used when the UCA responds to an external or internal message or event. Attach the enabling service to the UCA.

Screen Name: *Configuring a time elapsed undercover agent (UCA)*
Screen Number: *14*
Script:

After you create the UCA, you will see the details page. The attached service is listed, and the output variables of the enabling service are listed as an output of the UCA. The output data of the UCA is sent as part of the message payload. At the bottom of the details page, developers must create the time schedule that the UCA fires. When the enabled check box is checked, the UCA is now active and is following the time schedule.
When the UCA fires, the enabling service runs, and the output of the enabling service is passed as part of the message payload. Any message events that are listening to this UCA reacts to the message.
Time elapsed UCAs can also be run on demand by clicking the run now button.

Screen Name: *Implementing message start events*
Screen Number: *15*
Script:

The final step would be to implement the start event on a BPD. You must configure the start message event to work properly by designating the attached UCA. Then, specify any conditions on the processing of incoming messages. Finally, specify whether the UCA must consume messages.
Consider what happens when you use the consume message option. When a message is delivered to a running process, the first message event in the BPD that can accept it consumes the message. The UCA that is attached to the message event determines whether to consume the message. When a message is consumed, message event stops further message processing on that message. Any other message event in the BPD instance that can accept it cannot consume the message. If the execution of the BPD instance loops back and reaches the same message event, the message event is not processed. If a new instance of the message is delivered to the process instance, this message is available for consumption again.

The final option is the durable subscription. Durable subscription allows an incoming message to be delivered to the message event, even if the token does not reach the message event. This option is not available for message start events because the message start event creates the instance.

Screen Name: *Mapping message start event variables*
Screen Number: *16*
Script:

The final step would be to map the outputs of the start message event to the variables in your process. Again, the start event output variables are defined as outputs in the enabling service as part of the UCA message.

Screen Name: *Demonstration - UCAs and start events*
Screen Number: *17*
Script:

The following walkthrough is an instructor demonstration on implementing message start events, creating an enabling service, configuring UCAs, and triggering UCAs.
Click start demonstration to begin.

Screen Name: *Unit 9: Conducting playback 3*
Screen Number: *18*
Script:

Topic 3: Applying asset tagging. As library artifacts are created, organizing assets is a constant need throughout development. Favorites, tagging, and smart folders can be used to provide quick access to all of process application assets. Next, the details on how to apply asset tags to all development artifacts in the Process Designer library is covered.

Screen Name: *Creating favorites*
Screen Number: *19*
Script:

In IBM Business Process Manager, there are several ways to organize your assets. Many of these concepts might already be familiar to you.
Any asset can be marked for quick access by marking it as a favorite. All of these assets automatically show up in the smart folders category favorites in the lower left corner. Select the star next to any asset to mark it as a favorite

---

Screen Name: *Tagging library items*
Screen Number: *20*
Script:

Many of your assets fall into categories. You can use a pre-defined set of tags or create your own tags for groups of items. After you tag items, you can view them by asset type or by tagged items.

Right-click an asset and select Tag to tag an item. To view by Tag, select the Type menu and choose Tag.

Screen Name: *Smart folders*
Screen Number: *21*
Script:

Assets can also be organized into viewing folders. There are already some smart folders included by default. For example, Changed Today includes all assets that changed in the current day by any user that can access the process application. Smart folders work by creating rules to organize assets. You can include library items that are based on the presence of a particular tag, creation date, item type, or other criteria. Smart folders do not move your assets into other folders. You can categorize and view your current assets in a different way.

The smart folder decisions evaluate the rules from top to bottom. If no rules match existing library items, the smart folder is empty. Here are a few smart folder ideas that are good practices for organization: Top-level business processes, UCAs, web services, utility services, BPD wrappers, task services, and data access services. Share your smart folders that are standard - they are probably helpful to other designers. In some cases, the library already organizes your items. For instance, there is already a folder for Human services so you would not create a smart folder for coach services.

Screen Name: *Demonstration - Organizing assets*
Screen Number: *22*
Script:

The following instructor demonstration covers how to create favorites, how to tag library items, and how to create smart folders.
Click start demonstration to begin.

Screen Name: *Unit 9: Conducting playback 3*
Screen Number: *23*
Script:

Topic 4: Accessing and manipulating external data. A major concern of any enterprise system is its integration capability with other systems. For example,

you might want users to choose from a list of products available from a web service. In this playback, you focus on integrating with other systems. When you want to integrate, use an integration service. Integration services are in the library in the Implementation category. The integration framework allows interaction with existing applications, ERPs/CRMs, web services, and external data sources with a number of protocols.

Screen Name: *Integrating with other systems*
Screen Number: *24*
Script:

With the integration framework of Business Process Manager, you can interact with:

- Existing applications
- Enterprise Resource Planning (ERP)
- Customer Relationship Management (CRM)
- Web services
- External data sources through a number of protocols

Process Designer provides support for both outbound and inbound integration. When it communicates with an outside system to retrieve, update or insert data, it is called an outbound integration. Outbound and inbound integrations are implemented in Process Designer through integration services.

Screen Name: *Integration services*
Screen Number: *25*
Script:

When developers communicate with other systems from a process application, they do so by using an integration service and a Java or web service integration component. For Process Designer, the integration services are in the library in the implementation category. Integration services have two basic integration components to designate the method and protocol that is used to connect with outside services. When you consider which integration component type to build, think about the available integration methods and protocols:

- Web service integrations tend to be easy to build and are useful, especially if you are not passing volumes of information
- Java integrations are robust because they are built in Java, and sometimes you can find existing Java connections to existing systems

The integration components handle the lowest level of communication to the outside application or data source.

Screen Name: *Using integration service components*
Screen Number: *26*
Script:

Developers can add an integration service to multiple general system services. Developers can also use an integration service more than one time to produce different data from different sources. When you run an integration service at run time, it calls the web service or Java integration component, which in turn calls the external application. What you see in the integration service at run time is not the output from the external application, but the output from the integration component. That output is usually XML.

Screen Name: *Using integration service components*
Screen Number: *26*
Script:
You can add an integration service to multiple general system services, or even use it in the same service more than one time to produce different data from different sources. Integration services call either web services or Java integration components, and those services in turn message an external application.
The return from the integration is not the raw return from the integration. The integration service does not show the output from the external application. It shows the output from the integration component. That output is usually XML.

Screen Name: I*ntegration component for a web service*
Screen Number: *27*
Script:

The web service integration component is built by using the Apache Axis web Service Client framework, which is an implementation of SOAP. Web service integration components use the SOAP connection to callout to web services by discovering the ports and parameters from the WSDL document. The web service integration component hides the complexity of the underlying WSDL document, SOAP request, and SOAP response. It also converts the inputs into the appropriate XML and the outputs into the appropriate variables. When they design the data to pass in and out by way of web services, developers avoid circular references in data structures. Circular references do not map correctly to the WSDL document that is needed to define the web service.

Screen Name: *Web services*
Screen Number: *28*
Script:
A web service is one way for external systems to affect the flow of a process. The method for accomplishing this integration is to create and publish a web service endpoint through a Web Services Description Language (WSDL) interface. External applications can initiate a particular IBM Business Process Manager service or set of services with the WSDL.
In the Behavior section, the web service provides the WSDL URI. You are able to protect the web service with the Protected check box. You can also set the target namespace if wanted.

Screen Name: *Policy set support for web services*
Screen Number: *29*
Script:

The Security and Policy determine the type of security that developers use for the web service. When using a Policy Set, the list of policy sets shown depends on the policies available on the server. Some default application policy sets include: WSHTTPS default, WSAddressing default, and Username WS-Security default. Developers can also create extra application policy sets in the WebSphere Application Server administrative console.
The **Policy Binding** specifies the name of the general client policy set binding, which contains system-specific configuration parameters like user name and password information. Developers also create extra policy set bindings in the WebSphere Application Server administrative console.

Screen Name: *Simple configuration of inbound web service details*
Screen Number: *30*
Script:
Use an environment variable (found in the Process App settings page under the Server tab) for any settings that are specific to the different environments that the inbound web service is hosted on. More detail on environment variables is covered in the next topic.

Screen Name: *SOAP header support for inbound and outbound web services*
Screen Number: *31*
Script:
Sometimes SOAP services require custom data to be included in the header. IBM Business Process Manager supports setting SOAP headers in an outbound

web service call, and allows passing in SOAP headers through inbound web services. This support is provided through three mediums:

- Presupplied `SOAPHeaders` and `SOAPHeader` object types
- Variables to store the soap header data
- Through setting and reading those JavaScript variables automatically by the system

---

Screen Name: *SOAP header variable types in the system toolkit*
Screen Number: *32*
Script:
IBM Business Process Manager provides both `SOAPHeaders` and `SOAPHeader` variable types in the system toolkit. An implicit SOAP header is one that is not defined in the web service WSDL document. As part of the outbound web service integrations, developers add implicit SOAP headers to the web service request messages and retrieve SOAP headers from response messages.

Screen Name: *Java integration component*
Screen Number: *33*
Script:
As the name implies, use the integration component to integrate quickly with any Java API by calling a class method that uses reflection. This extensible integration framework can interface with most third-party Java APIs, allowing it to support many integration scenarios. Single Java integration components can reference, or call, one method of the class. To call a different method within the class, you need to create an integration service with a separate Java integration component. Before you create an integration service with a Java integration component, the Java class and method must be present in the library. By default, the classes in the Java package are available in the integration JAR file, which is included in the system data toolkit. Authors can add their own custom-built JAR files either directly in the process application or through a toolkit. JAR files that are included in a toolkit can be shared between process applications.

Screen Name: *Pre-built integration services*
Screen Number: *34*
Script:

IBM Business Process Manager contains several pre-built integration services that already contain integration components. These integration services support common database interactions, including support for parameterized queries. In

addition, these services can automatically map query results directly into a wanted variable type.

One example is the integration service named SQL Execute Statement.

Screen Name: *Demonstration - Pre-built integration services*
Screen Number: *35*
Script:

The following instructor demonstration views the pre-built integrations services that are contained in the system toolkit.
Click start demonstration to begin.

Screen Name: *Unit 9: Conducting playback 3*
Screen Number: *36*
Script:

Topic 5: Exposed process variables (EPVs) and environment variables (ENVs). Both the environment variable and the exposed process variable can be used as a “global” variable inside of a BPD.

Screen Name: *Environment variables (ENVs)*
Screen Number: *37*
Script:

IBM Business Process Manager has capacity to define environment variables. Take advantage of these environment variables. EPVs ensure that your process implementations are using correct values no matter which environment you deploy to.
For example, suppose that your process includes an implementation that requires the port number for an external application. By using an environment variable, you can set the port number for each environment in which the process is run. A running process on a test environment might have a different port number than the same running application on a production environment. Developers can specify a Default value and a value for each type of Process Server. There is a limitation of four environments (plus a default value) for environment variables. If you need more than four environments, consider an exposed process variable for each environment needed.
When you name environment variables, start with lowercase letters and separate the words with periods.

Screen Name: *Exposed process variables (EPVs)*

Screen Number: *38*
Script:

Another variable-type that is used in BPDs and services is an exposed process variable (EPV). EPVs allow business users to modify a value of a variable without providing access to the IBM Process Designer. A business user can log in to the Process Administration Console and change the value of an EPV. The user must be authorized to change the EPV.

Screen Name: *EPVs versus environment variables comparison*
Screen Number: *39 - 40*
Script:

The table that is shown gives an overview of the advantages and disadvantages of each type of variable. This table is not an exhaustive list of advantages and disadvantages. It is intended to provide a foundation when you consider which type of variable to use.

Screen Name: *Demonstration - ENVs vs. EPVs*
Screen Number: *41*
Script:

The following instructor walk-through demonstrates defining environment variables and defining exposed process variables.
Click start demonstration to begin.

Screen Name: *Exercise 14 – Creating playback 3 assets*
Screen Number: *42*
Script:

This exercise comprises the step-by-step process of creating a service to query a database and populate a list.

After completing this exercise, you should be able to:

- Create a decision service
- Create and configure a UCA
- Start a BPD with a message start event

- Use tagging to organize your assets
- Query a database to obtain information and populate a list variable
- Create environment variables (ENVs) and exposed process variables (EPVs)

Screen Name: *Unit summary*
Screen Number: *43*
Script:
Playback 3 is now ready. The goal is to demonstrate “real data, real time.” Authors log on to the Process Portal and demonstrate the integrations on which the process application depends. The business rule logic and tasks that are being created as a result of the decision service are shown as well. Authors also demonstrate message events in the BPD that affect process flow in this playback validation session. Some of the integrations that are created in playback 3 can enable coach functions. Those integrations are demonstrated in the next playback validation session.

Having completed this unit, you should be able to:

- Create a decision service
- Create a message start event
- Create an enabling service
- Create and configure an undercover agent (UCA)
- Start a BPD with a message start event
- Organize assets with favorites, tagging, and smart folders
- Define the basic function of an integration service
- Identify the components of the IBM Business Process Manager integration architecture
- Describe how integration components interact with services
- Configure and define integration services for outbound integration
- Describe the differences between an environment variable and an exposed process variable
- Identify the components of the IBM Business Process Manager integration architecture
- Describe how integration components interact with services

- Configure and define integration services for outbound integration

---

Unit 10: Playback 4

Screen Name: *Unit introduction*
Screen Number: *1*
Script:

Playback 4 returns to the coaches created in Playback 2 to enhance them. The added functions and interactivity are intended to create a better user interface. It is important to provide users an intuitive interface so they can accomplish their tasks within the shortest amount of time. This playback usually garners the most attention from stakeholders, so be prepared to stay on this playback phase until you get the wanted presentation.
As you customize your coaches, make sure that you consider the effect to the project timeline. Sending enhancements to the backlog and sacrificing presentation elements to adhere to timelines might be necessary. Talk to project management and consider the best way to approach these enhancements or to compromise as necessary.

Screen Name: *Unit objectives*
Screen Number: *2*
Script:

After completing this unit, you should be able to:

- Create tabs on a coach
- Add a visibility rule to an input control
- Apply a class to a control
- Change the look of a control through CSS
- Create a coach view

Screen Name: *Topics*
Screen Number: *3*
Script:

This unit consists of the following topic:

- Enhancing Coaches

Playback 4 provides the opportunity to add scripts, interaction, stylization, menus to select from, and all of the other user interface enhancements. These enhancements are necessary so that process participants can complete their tasks.

When you approach playback 4, avoid adding extra functions when it does not provide value. Now is not the time to cause a regression error; make sure that everything that worked in a previous playback still works as expected.

Screen Name: *Basics of coach enhancement*
Screen Number: *4*
Script:

Coaches use the same technology as any web development platform to serve web content to users. Coaches are composed of HTML, JavaScript, and CSS. The three elements control the content, functions, and presentation of every webpage.
Coaches are made of HTML divs and spans that control the structure of the page. The elements contain controls and labels. The stock JavaScript and CSS files are linked through <include> elements in the HTML, and served from the IBM Business Process Manager server.
Although coaches come pre-loaded with CSS and JavaScript to control the stock functions, any you can accomplish any presentation wanted. That includes creating custom HTML controls from scratch, creating custom style sheets, and changing or adding JavaScript functions. Any web 2.0 elements can be included in a coach.
Some organizations might be hesitant to develop coaches if they are unaware of the capabilities. It might surprise you to know that there are no limitations to using coaches to create your WebPages. If it can be built and shown in a browser, it can be done with a coach.

This unit focuses on using controls that are already included in the IBM Business Process Manager coach toolkits that use HTML, JavaScript, and CSS. These elements enhance the stock coaches, but if these stock controls do not meet your needs, feel free to create your own.

Screen Name: *JavaScript Libraries*
Screen Number: *5*
Script:

To assist your development, any JavaScript library can be used. Developers can use the Dojo library that is included by default on every coach. If you are familiar with another library, feel free to add a Custom HTML block and include the library on your coach.

Screen Name: *Coach tabs*
Screen Number: *6*
Script:

Large coaches have numerous disadvantages. Large coaches affect coach performance. Users might sit around waiting for the entire page to load. Or, the user might become confused or overwhelmed when you work with such a large amount of data. Similar fields next to each other might not provide a user enough contextual information about what information is input into the fields. One solution is to group similar data into tabs and provide a wizard-like interaction with the forms the user must complete. The coaches become more useful and the overall user experience is improved.

It is easy to add tabs to any coach without having to add custom HTML and JavaScript to enable the tabs. Click and drag a tabs control onto the palette and move the sections into the individual tabs. All of the show and hide JavaScript functions are provided to the developer through the control.

Screen Name: *Demonstration - Group controls into tabs*
Screen Number: *7*
Script:

The following walkthrough is an instructor demonstration on grouping controls into tabs on a coach.
Click start demonstration to begin.

Screen Name: *Implementing a select control*
Screen Number: *8*
Script:

The coach that you built in Playback 2 contained input boxes where a user might insert any type of data into an input box. One way to standardize the input data is to provide options for a user to choose from. There are many different options to allow a user to choose from a list of choices on a coach. The intent is to limit the values that a user can input for a certain control. One common approach is to provide a select menu of options for the user to choose from. The control itself is bound to a variable, similar to the input box used in Playback 2. But now, the selection data comes from a list that is retrieved from a database.
The easiest way to implement the control is to store the dynamic data in a name-value pair (NVP) variable type. This NVP stores the selections that a user can choose from. This example uses tw.local.departmentList to store a list of department names as the NVP name, and the department number as the NVP value. The step before the coach retrieves the data with a SQL integration service you created in playback 3, and saves the data in the departmentList object.
Drag the integration service onto the palette and hook up the flows. Map the output of the integration service to a local variable (in this case, tw.local.departmentList). This variable now feeds the control on the coach.

Screen Name: *Implementing a select control*
Screen Number: *9*
Script:

The next step is to identify the input control that is created in Playback 2, and change it into a select control. The coach bonus toolkit provides a nice looking select box. The view option is what controls what the control acts and looks like. A view is explained in more depth later on in the unit. Change the view from the Text view to the Single Select from the coach bonus toolkit. The image on the palette immediately changes to reflect the new look of the select box.
There is no need to change the binding of the control because the option selected by the user is assigned to the same variable.

Screen Name: *Configuring the select list*
Screen Number: *10*
Script:

The final step is to tell the control what values must be displayed to the user. You must implement what values must be stored in your bound variable when the user makes the selection. By selecting your list of name-value pair, the department list allows only those options in list to be selected from.
When the user selects their choice, the Binding Value of their selection, in this case tw.local.departmentList.value, must be bound to a new private variable of the same type as the list. Then, after the coach, you must map the .value of the new private variable to the variable tw.local.hiringRequestData.requisitionDetails.generalDetails.department. That last step will happen after the coach, and places the data back into the business object.

Screen Name: *Demonstration - Implement a select control*
Screen Number: *11*
Script:

The following walkthrough is an instructor demonstration on changing a text control to a single select control.
Click start demonstration to begin.

Screen Name: *Adding dependent visibility to a coach field*
Screen Number: *12*

Script:

When adding coach views elements to a coach, all the fields are visible by default, set to Same as parent. This applies the visibility of the parent section to this element. To apply a static value to the visibility of the element, select the value radio button under source, and then select the visibility from the drop-down menu.
Editable is the most common setting, where users are able to add, delete, or change the value of the input box.
Required is the same as editable, but requires a value to be entered into the control. The user is not able to post the page and move away from the coach down a flow on the service without an entry in the field. The required setting only checks whether a value was entered, but does not validate the entry. If you require the entry to meet a rule or restriction, you must implement a validation service on the control.
Read only allows users to see the control but they cannot change the value.
Hidden and none values tell the browser not to render the control on the screen. The difference is hidden reserves a space in the layout, so a blank area is shown on the screen in the layout. None collapses this section on the coach when rendered.

Screen Name: *Setting dynamic visibility through a script*
Screen Number: *13*
Script:
Consider the scenario when a user selects a specific value in a drop-down box, they must then provide an explanation in a separate box. The explanation box must be visible only when that specific value is selected, and hidden when any other selection is made. You cannot use a static selection as the source for the visibility.
If the visibility of a control is dependent on the input of a different field, you can use a script or rule to determine the visibility of the control.
Using the script option as the source of the visibility provides the greatest amount of control over your visibility. Select a target variable to watch, and input JavaScript code to be run when the variable changes value. Most coach controls allow a variable to be used instead of the selection from a selector box.
To use this approach, the control that the show or hide function is dependent on must be bound to a value listed under the variables tab in the coach. The function must return the visibility of the control as a string.

Screen Name: *Using a visibility rule*
Screen Number: *14*
Script:
Creating a visibility rule is similar to building a rule service. You can set visibility rules for either a variable value or a team membership. You can add multiple rules, but the rules have an “or” relationship between them. They also are

evaluated from top to bottom, and the first rule that evaluates to true is applied. If all other conditions evaluate to false, the final condition provides the default value.

Screen Name: *HTML attributes*
Screen Number: *15*
Script:
The HTML Attributes section in the Properties tab is where you override styles for a control. It includes adding any CSS classes to the object or any attributes added to the control's HTML tags.

Screen Name: *Applying CSS to a control*
Screen Number: *16*
Script:
A CSS (Cascading Style Sheet) class, can be added to any control that uses the HTML attributes. This class is added to the DIV that contains the control.
The first step to changing the look of a control is to apply a class to the control as shown in the image on the top. Next, add a customHTML section onto the canvas, and apply a style that targets the class that is applied to the control. Because the class is applied to the outer div of the control, you must select a child member of the control in order to target the input box control. Otherwise, your class is applied only to the outer div. In the example, to apply a width attribute to the input control, you select the object using .largeInput .dijit and then the textbox element.
After that customHTML section is debugged and passed cross-browser testing, you can move the style into a coach view to be reused with other objects that require the same styling. You might also create a new coach view that includes the new styling to be shared with other coaches and used as a specialized input control for other coaches.
If the CSS is applied to multiple coaches, consider moving the CSS to a CSS file and include the CSS style sheet on every coach or coach view. Promoting code in this fashion provides a structured approach to deploying organizational styling to your pages.

Screen Name: *Demonstration - Dependent visibility*
Screen Number: *17*
Script:

The following walk-through covers:

- Viewing the coach bonus toolkit
- Assigning a variable to a control option
- Implementing dependent visibility on a coach

Click start demonstration to begin.

Screen Name: *Unit 10: Conducting playback 4*
Screen Number: *18*
Script:
Topic 2: Coach views.

Screen Name: *Coach views*
Screen Number: *19*
Script:

You applied the CSS and achieve the wanted style. Now, create a reusable asset that can be shared with multiple coaches or even multiple process applications. By moving the class into a coach view, the class can now be a shared asset. Objects like the tabs component are made from a coach view. You already worked with coach views, but now is your chance to create your own.
Coach views are reusable sets of user interface that users use to interact with a business object or service. Coach views can consist of one or more other coach views, data bindings, layout instructions, and behavior.
Because coach views are reusable, coach views and coaches can share parts of their user interface with other coach views and coaches. For example, you create a coach that has a coach view that contains a set of address fields. If you create a second coach that needs address fields, you can reuse the coach view from the first coach. In both cases, the coach is using an instance of the coach view. You can edit the properties of each instance independently. For example, changing the label of one coach view instance does not change the label for the coach view. It also does not change the label or any other instances you use of the coach view. Both instances of the coach view use a reference to point to the coach view definition. This approach means that if the coach view master changes, you can see the change reflected in every instance of the coach view.
You can create a coach view in the process application or in a toolkit. In general, create highly reusable coach views in toolkits, and customize the coach views in process applications.
If someone edits a coach view in a toolkit, the changes apply to every application that used the coach view in the toolkit. Because editing a coach definition can affect many instances, be careful in your changes.
You cannot directly edit the definition of the coach view from within the parent coach or coach view. Instead, you must open the coach view definition first before you can change it.

Screen Name: *Create a coach view*
Screen Number: *20*
Script:

Similar to creating a human service, you can create a coach view with the user Interface category.

Screen Name: *Coach view definition page*
Screen Number: *21*
Script:
When you open a coach view definition to edit it, you see tabs to configure the overview, behavior, variables, and layout of the coach view.
The overview shows:

- The coach view name
- Information about the coach view
- The images that are used to represent the coach view during design time
- How the coach view is used

You can also tag your coach view to make it easier to find in the library and on the palette.
The variables tab displays:

- The bound business data
- Configuration options (which includes Ajax services)
- Localization resources available to the coach view or used by the coach view

Finally, the layout tab displays:

- The coach views
- Controls that are contained within the coach view
- Their relative positions
- The palette, which contains items that you can add to the coach view

---

Screen Name: *Defining coach view behavior*
Screen Number: *22*
Script:
The behavior tab displays the scripts and CSS files that are used by the coach view. The Behavior page is also where you define event handler code. The Event handlers are the entry points for the code of the coach view. While the coach view might reference supporting JavaScript files, the event handlers contain the functions that the IBM® Business Process Manager framework calls.
Coach views can be stock or custom. IBM BPM provides the stock controls, which are coach views. You can find them in the Control category or in the

Section category on the palette. Custom coach views are ones that you create or the views that are provided by other programs or companies. In terms of use, IBM BPM treats stock and custom coach views identically.

Screen Name: *Demonstration - coach views*
Screen Number: *23*
Script:
The following demonstration covers creating a coach view.
Click start demonstration to begin.

Screen Name: *Exercise15 – Enhancing a coach*
Screen Number: *24*
Script:

This exercise comprises the step-by-step process of conducting the playback 4 of a BPD.

After completing this exercise, you should be able to:

- Create tabs on a coach
- Change a text control to a single select control
- Add a visibility rule to an input control
- Apply a class to a control
- Change the look of a control through CSS
- Create a coach view

Screen Name: *Unit summary*
Screen Number: *25*
Script:

Playback 4 is now ready. Show off all the new features you built into your coaches. Log in to the Portal and demonstrate how each coach looks to the different participants in your process. This playback garners intense scrutiny, so before you enter this playback session, make sure that no regression errors occurred. All the coaches must function correctly, and this playback shows off the enhanced features you created.

Business users always demand heavy user interface (UI) requirements. It traditionally is a struggle between developers and project management. Talk

about the compromises to the UI. A consensus can be built around what is necessary to make the job of a participant easier to accomplish in the process. The level of effort can be balanced with spending a tremendous amount of time to create complex UI screens, which might hold little business value.

Having completed this unit, you should be able to:

- Create tabs on a coach
- Add a visibility rule to an input control
- Apply a class to a control
- Change the look of a control through CSS
- Create a coach view

Unit 11: Conduct Playback 5

Screen Name: *Unit introduction*
Screen Number: *1*
Script:

The process application is fully functional and implements all of the requirements for the process, but what happens when unexpected things happen? System errors occur suddenly. Unexpected user input causes an instance to fail, or web services that seemed reliable are suddenly unavailable.

What happens then?

You are not able to harden your process to handle every situation. But, building in some logic to handle the things most prone to error is a good way to approach hardening. Spending your time on the 90% solution provides better hardening for your process. Do not spend your time on the most remote errors that might occur.
Playback 5 now concentrates on hardening your process. Building in robust error handling is necessary, and developers must learn to expect the unexpected. Now is the chance to ask, “How do you handle things if the process does not run as expected?”
This playback most likely requires changes to the BPD, something that was largely untouched since Playback 0/1. Make sure any changes that you make do not break anything currently functioning correctly. Breaking items that were running correctly runs counter to the point of error handling.

Screen Name: *Unit 11: Conducting playback 5*
Screen Number: *2*
Script:

After completing this unit, you should be able to:

- Catch an error in a BPD and service

Screen Name: *Topics*
Screen Number: *3*
Script:

This unit consists of the following topic:

- Hardening a process
- Validating error handling

Screen Name: *Handling errors in BPDs*
Screen Number: *4*
Script:

When a process is developed in Process Designer that includes integrations, authors need to anticipate potential system exceptions. Authors then create the components that are required to handle those exceptions when they occur. For example, if a BPD integrates with a database system, that database is not always available when each new instance of the BPD runs. So, when authors develop the integration, they need to build the ability to detect errors and recover in a predictable manner into the exception-handling process.
When modeling error handling as part of your business process definitions (BPDs), you can catch errors with error intermediate events or event subprocesses. You can throw errors with error end events.
Exception handling is a topic to talk with system administrators, developers, and other interested stakeholders to plan which actions are taken when the system has errors.

Screen Name: *Catching errors*
Screen Number: *6*
Script:

Use error events to specify how errors are thrown and caught in your runtime environment.
You can assign error codes and error data to errors that the error end event throws.

Errors are caught in the following order in your runtime environment:

- The boundary events catch errors that the attached activity raises.
- Errors are caught in the error event subprocesses if the subprocess is in a BPD or in an unattached intermediate error event in a service
- Errors are propagated to the next level.

Screen Name: *Unit 11: Conducting playback 5*
Screen Number: *6*
Script:

Topic 2: Handling errors in services. Processes that you develop in IBM Business Process Manager might include integrations with external systems, server scripts, and other complex implementations. You need to anticipate potential system exceptions and create the components that are required to handle those exceptions when they occur. For example, if a BPD integrates with a database system, that database might not be available with each new instance of the BPD. When you develop the integration, build in exception handling to detect errors and recover in a predictable manner at the service level.

Screen Name: *Catching errors in services*
Screen Number: *7*
Script:

Similar to BDs, you can use error intermediate events to catch errors, and you can use error end events to throw errors. Attach error intermediate events to steps in your service to catch errors in that step. You can also include error intermediate events to act as global error handlers in the service flow.
Determine whether errors can be handled immediately, and normal processing can continue, or if another error can be thrown at another level. Then, implement error handling from the bottom up.
Use an error end event to throw a specific error, and specify error codes and error data for the error.
Consider specifying the error data to catch specific errors. For example, you might filter on the error code for the types of errors that are caught. Then, map the error code to a variable after the errors are caught. When all errors are caught, or if only an error code is specified, the error data is captured in an XMLElement in the tw.system.error variable.
Be sure to include the appropriate logic and an error end event in your service. Throw errors when they still exist after attempts to handle them. Throwing the errors ensures that the errors are passed to parent processes and services and can be handled at a higher level.

Screen Name: *Demonstration - Error handling*
Screen Number: *8*
Script:

The following walkthrough is an instructor demonstration on hardening a process application in Business Process Manager. This demonstration covers catching errors in both a BPD and a service. Click start demonstration to begin.

Screen Name: *Exercise*
Screen Number: *9*

Script:

This exercise is the step-by-step process for attaching a catch exception component for a service.
After completing this exercise, you should be able to:

- Harden a service with a catch exception component

Screen Name: *Unit summary*
Screen Number: *13*
Script:

This playback is the final one. The goal of this playback is to demonstrate what happens when your process encounters an error, and to demonstrate how that error is handled. Demonstrating error handling might take some test harnesses to generate the errors internally.
Because your processes and the sheer number of artifacts that are created might be large, demonstrating all of the error handling might be overwhelming.
Consider demonstrating the different error handling patterns that are used in your process application. Stakeholders can get a good idea of the general way errors are handled when they occur. Convey the general principles that were used for error handling without having to demonstrate every instance of error handling in your process app.
Consider adopting a community developed toolkit that is called the “General Exception Handling Toolkit (GEX)” on the IBM Business Process Manager Wiki. This toolkit can be a starting point toward your own organizational error handling framework.
Having completed this unit, you should be able to:

- Catch an error in a BPD and service

ZB808 Course Summary

Screen Name: *Course wrap-up*
Screen Number: *1*
Script:

This unit completes the Process Modeling and Implementing with Business Process Manager course. This unit provides summary material for the course and information you can use to continue with Business Process Manager training.

Screen Name: *Course wrap-up: Course summary*
Screen Number: *2*
Script:

You can now:

- Describe why process modeling is an important phase in the BPM lifecycle
- Identify how to use Process Designer to create a process application
- List and identify the core elements that are used to create a BPD in Process Designer
- Translate workflow steps into business process activities and nested processes
- Use gateways to control the process flow
- Validate that the process model meets playback zero goals and requirements
- Identify how intermediate events are used during the execution of a business process
- Describe the architecture of IBM Business Process Manager
- Organize process assets into toolkits
- Manage variables and data flow
- Implement timer events
- Implement gateways and routing to control process flow
- Build a business data model
- Build services and user input forms (coaches)
- Create a snapshot for deployment
- Create a decision service
- Model and implement message events
- Apply asset tags to organize artifacts
- Enhance coaches for a rich user experience
- Implement effective error handling in processes and services

---

Screen Name: *Course wrap-up: Additional learning*
Screen Number: *3*
Script:

To learn more on this subject, see the websites that are listed, such as the WebSphere Education website. Send an email to the address listed on the screen for more course information.

Screen Name: *Course wrap-up: References*
Screen Number: *4*
Script:

The listed items are reference materials that are available for this course.

Screen Name: *Course wrap-up: Evaluation*
Screen Number: *5*
Script:

Complete a survey for this course by clicking the online survey link. WebSphere Education uses comments about this offering to constantly improve the course. In addition to completing the survey, a request for a certificate of completion for this course is available.