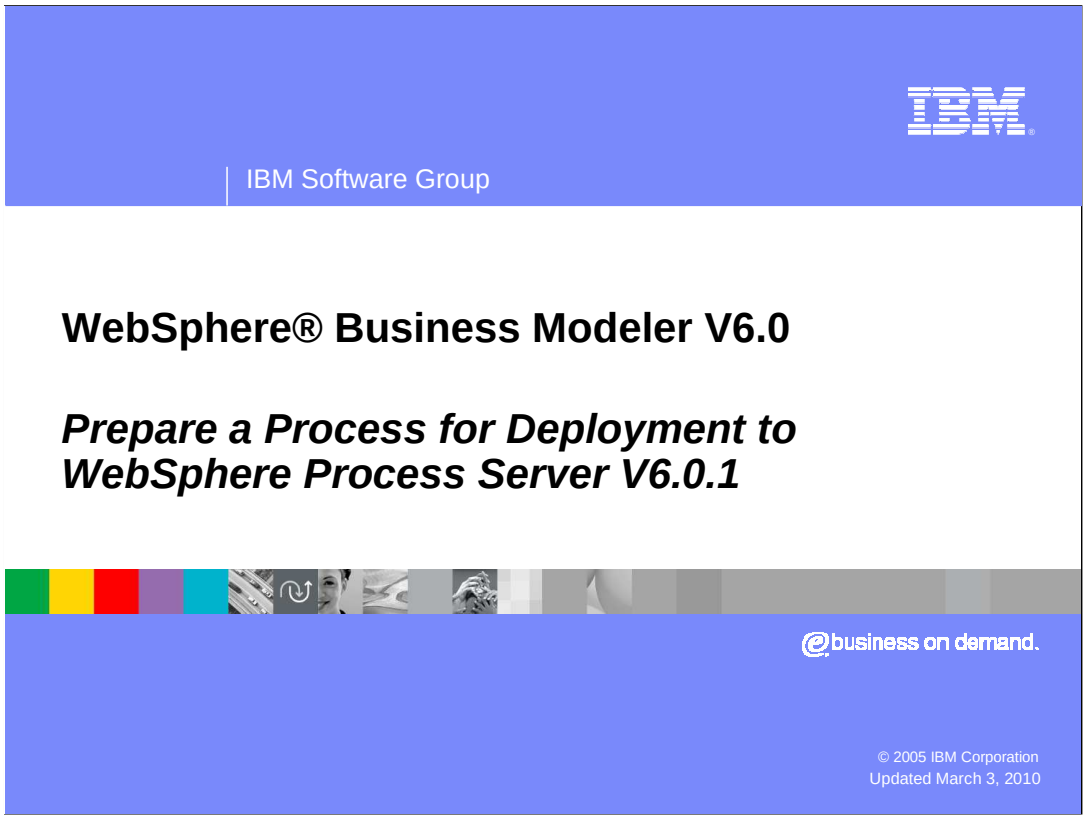


This presentation will provide an overview of preparing a Process in WebSphere® Business Monitor V6.0 for deployment to WebSphere Process Server V6.0.1.

Goals

- Explain the steps and options for preparing a process in WebSphere Business Modeler V6.0 for export for WebSphere Integration Developer V6.0.1
- Explain the steps and options for preparing an application in WebSphere Integration Developer V6.0.1 for deployment to WebSphere Process Server V6.0.1

The goals of this presentation are to discuss the steps and options available in WebSphere Business Modeler V6.0 prior to exporting it to WebSphere Integration Developer V6.0.1 and explain the changes that you can make in WebSphere Integration Developer V6.0.1 to map and connect the business process to an actual service at the IT level.

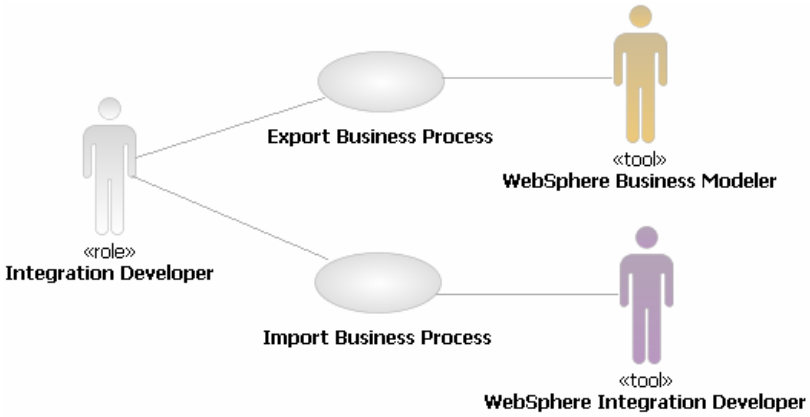
Agenda

- Overview
- Preparations in WebSphere Business Modeler
- Customizing in WebSphere Integration Developer
- Summary

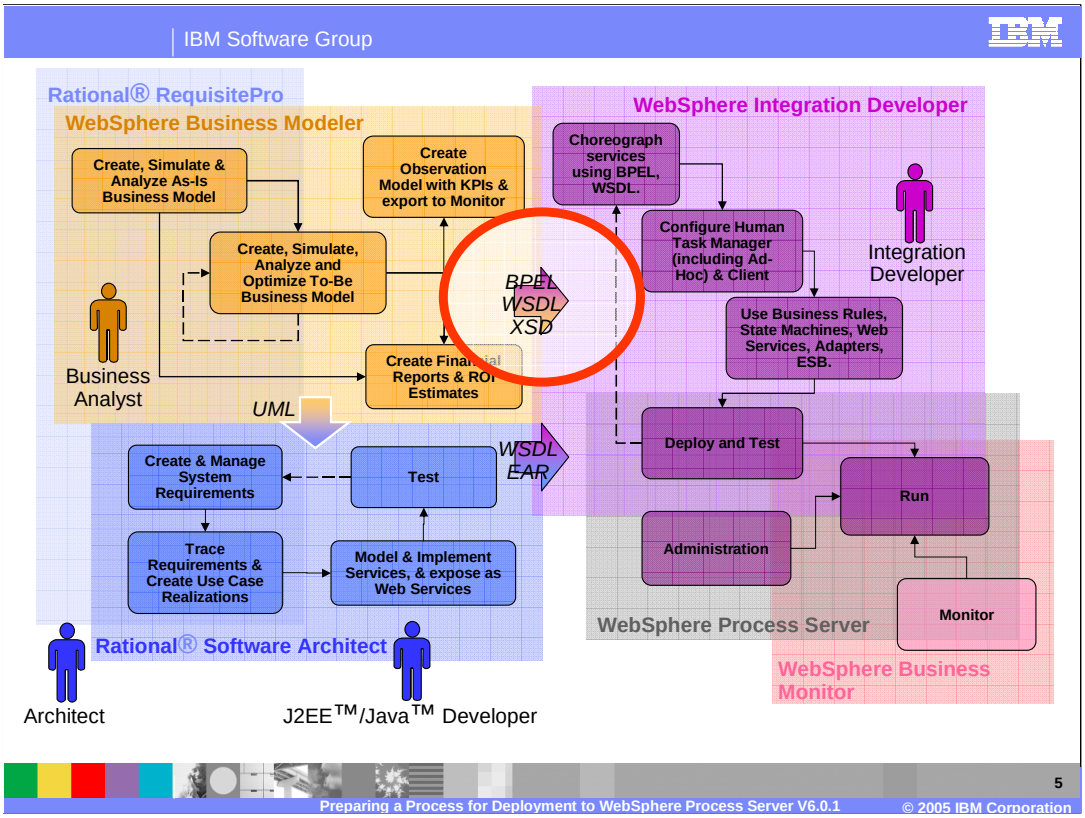
This section of the presentation will provide an overview of the series of steps required to move a process from WebSphere Business Modeler to WebSphere Integration Developer in preparation for deployment into WebSphere Process Server.

Preparation Overview

- Integration Developer focuses on connecting the business process and technical implementation



Business analysts are the primary user of WebSphere Business Modeler as they define the business process and perform simulations. However, preparing the business process for the production environment and customizing the process in WebSphere Business Modeler requires someone more familiar with the IT environment. An Integration developer or specialist specifies technical details to simplify the assembly of the process prior to importing into WebSphere Integration Developer, where assembly and implementation of the process is completed.



Several steps occur as a process is moved into the production environment. First, a business analyst will design the business process, simulate and optimize it using WebSphere Business Modeler. Once the process is complete, there are two types of exports that can be done from WebSphere Business Modeler.

It can be exported as a Unified Modeling Language (UML) document and consumed using Rational® Software Architect, which includes Rational Application Developer. For those services that are not implemented already, an architect and J2EE™/Java™ Developer can create the services to provide technology implementations and make the service interfaces available to WebSphere Integration Developer.

In WebSphere Integration Developer the service information is used along with the process, which is imported into the workspace. The activities in the process are connected to the services that were created by the architect and J2EE/Java developer or implemented with other components. Other types of integration logic can be added through business rules, state machines, Web services, and adapters to Enterprise Information Systems.

After creating the services and implementing the business process, the process can be deployed to WebSphere Process Server. If business metrics and key performance indicators were declared for the process in WebSphere Business Modeler, it can be monitored with WebSphere Business Monitor once it is deployed. The focus of this presentation is to look at the steps required to prepare a process for export from WebSphere Business Monitor and import it into WebSphere Integration Developer.

Agenda

- Overview
- Preparations in WebSphere Business Modeler
- Customizing in WebSphere Integration Developer
- Summary

This section will provide an overview of preparation for the transition between WebSphere Business Modeler V6.0 and WebSphere Integration Developer V6.0.1.

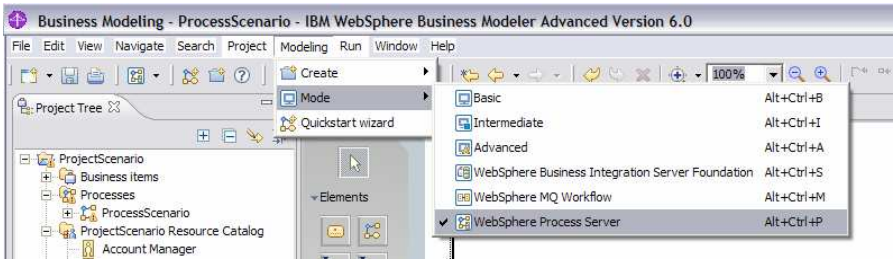
Preparing for Export – Overview of Steps

- Set WebSphere Process Server mode
- Customize process and nodes
- Export project (process and artifacts)

There are some steps that must be done prior to exporting the project from WebSphere Business Modeler. The first step is to set the WebSphere Process Server Mode to Modeler as soon as possible, followed by customization of the process and nodes, and finally, exporting the project. Each of these steps will be described in more detail.

WebSphere Process Server Mode

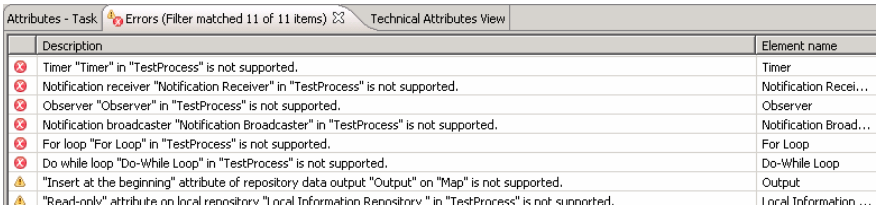
- WebSphere Process Server mode provides technical customization options and identifies unsupported elements
  - Applies a set of validation rules to the Advanced Modeling mode
- New technical attribute options available
  - Technical Specification Tab on process, service, and global task
  - Technical Attributes View for process and nodes
- Some modeling elements not available and disabled in editor
  - Notification Broadcaster and Receiver, Observer, Timer, Global repository, Do-while loop, For loop



Changing to WebSphere Process Server mode makes some options available and enforces additional sets of validation rules in addition to those rules that are already being enforced. Once this mode is set, a Technical Specification tab will be added on the Process Editor. There is also a Technical Attributes View for individual nodes as well as for the process. You will also find a Technical Specification Tab on the service and global task editors. In the Technical Attributes View, specific technical information or values for deployment for these nodes can be specified. With the WebSphere Process Server mode set, some of the modeling elements such as notification broadcaster and receiver, observer, timer, global repository, do-while loop, and for loop will be disabled in editor. so you will not be able to add these to a process. If these unsupported elements already exist in the process editor, they will be flagged as not compatible with WebSphere Process Server mode. It is up to you to decide if they should be replaced with elements that are supported in WebSphere Business Modeler or handled in WebSphere Integration Developer. Because unsupported elements cause additional work prior to deployment into production, it is recommended that you set the WebSphere Process Server mode as soon as you select WebSphere Process Server as your production environment.

Resolving Errors

- Some errors must be resolved; others may be ignored
- Errors to resolve
  - Errors regarding invalid interfaces
    - Each input criteria must have an output criteria
  - Connections to upstream nodes not allowed
    - Use While loop construct
- Unsupported elements can be ignored and resolved in WebSphere Integration Developer

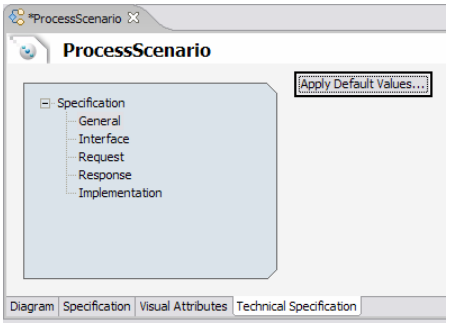


In addition to unsupported elements, you could also receive errors on other artifacts, represented by flags in the Error View. For the flags related to those unsupported elements, it is up to you to decide whether or not to keep those artifacts and move them to fix in WebSphere Integration Developer or change them to types of artifacts supported by Modeler. Other errors, such as invalid interfaces or error messages regarding input/output criteria should be resolved in within Modeler. If there is any connection back upstream within a process, that should be converted to a while loop in Modeler. As a best practice, be sure to make backups of the processes in a repository before making changes to resolve the flags.

Customizing Technical Attributes - Overview

New V6

- Technical Attributes are a set of runtime attributes related to specific business process elements (process, task, decision, loop)
- Technical Attributes are organized into sections (tabs)
  - General
  - Interface
  - Request
  - Response
  - Implementation
- Default values can be generated for all elements in a process
  - Implementation values not set
- Technical Attributes are optional
  - Default values will be generated at export time if technical attributes are not specified



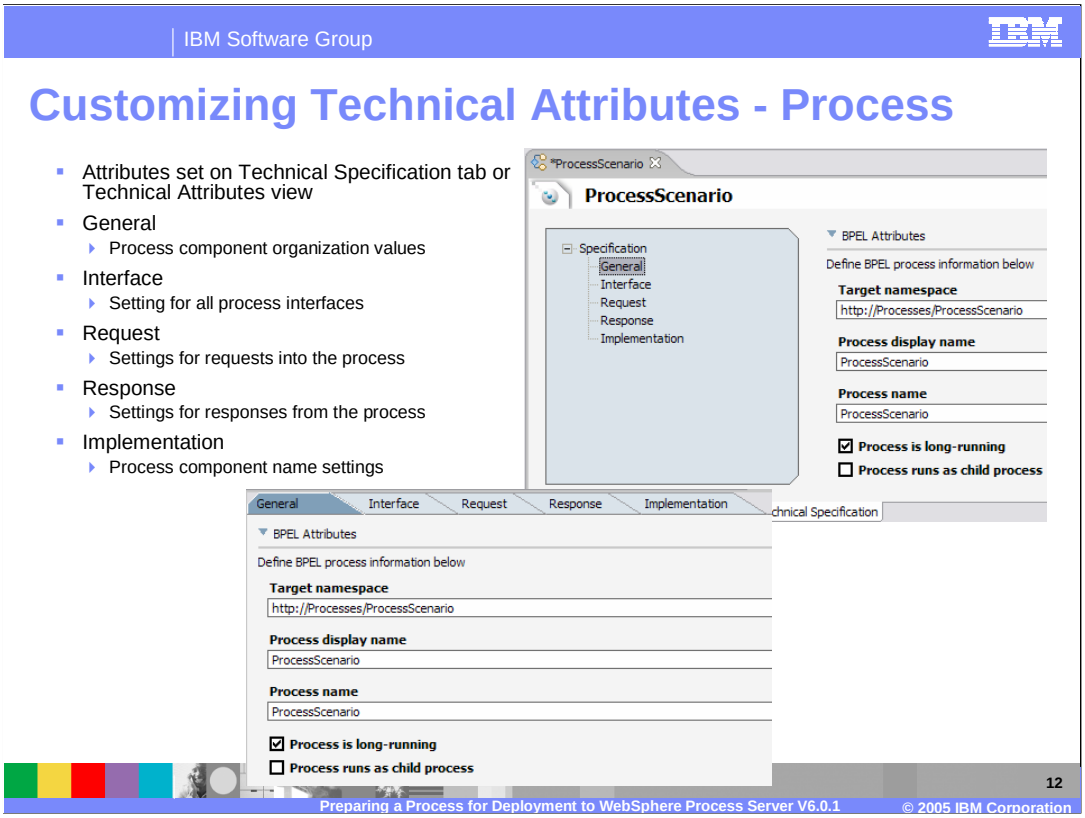
Enabling the WebSphere Process Server mode provides additional Technical Attributes that can be specified on processes and tasks depending on the type of element. Technical Attributes are organized into the General, Interface, Request, Response, and Implementation sections. Different attribute can be specified for each of these sections. You can generate default values for these attributes by selecting the Apply Default Values button. This provides a convenient way to set values in the correct format and allows you to easily update them with more appropriate and specific values. Technical Attributes are optional and if none are specified, default values will be used at export time.

Technical Attributes

- Certain tabs available for specifying technical attributes on different elements

		Tabs				
		General	Interface	Request	Response	Implem.
Element	Process	yes	yes	yes	yes	yes
	Global Task	no	yes	yes	yes	yes
	Service	no	yes	yes	yes	yes
	Local Task	no	yes	yes	yes	yes
	Global Process/Task/Service within a Process	no	no	yes	yes	no
	Decision	yes	no	no	no	no
	While Loop	yes	no	no	no	no
	Local Process	yes	no	no	no	no
	Local Repository	yes	no	no	no	no

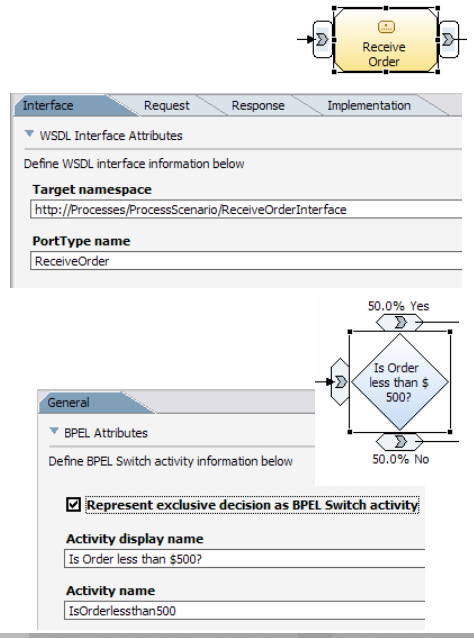
This table summarizes the technical attributes that can be specified for different elements. For example, General information can be specified for the process and decision and while loops. Interface information, including the Request and Response can be specified for elements as well. This information becomes part of the WSDL artifact generated for the element.



The technical attributes can be specified in multiple places for the process. On the Technical Specification tab in the process editor, the values can be set for the process, interface, request, response and implementation. These same values can also be set on the Technical Attributes view when the Diagram tab is selected in the process editor. Changing the values at either location will result in the same value being updated.

Customizing Technical Attributes - Nodes

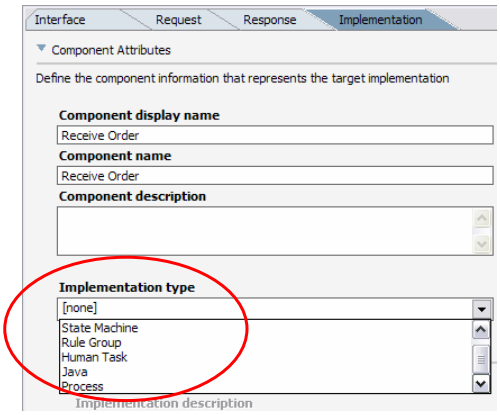
- Attributes set only in Technical Attributes view
- General
  - Specific setting for certain activities
- Interface
  - Settings for activities
- Request
  - Settings for requests message
- Response
  - Settings for response message if request/response



Different technical attributes can be specified for different nodes. Depending on the task, different technical attributes will also be available. For example, the Receive Order task has four technical attributes - General, Interface, Request and Response. Interface indicates Namespace and port type for the interface that defines the task. Request and response are the settings for request and response messages. For decision elements, a BPEL switch activity can be specified to be used. Otherwise, it uses logic in links inside a flow activity by default.

Setting Node Implementation

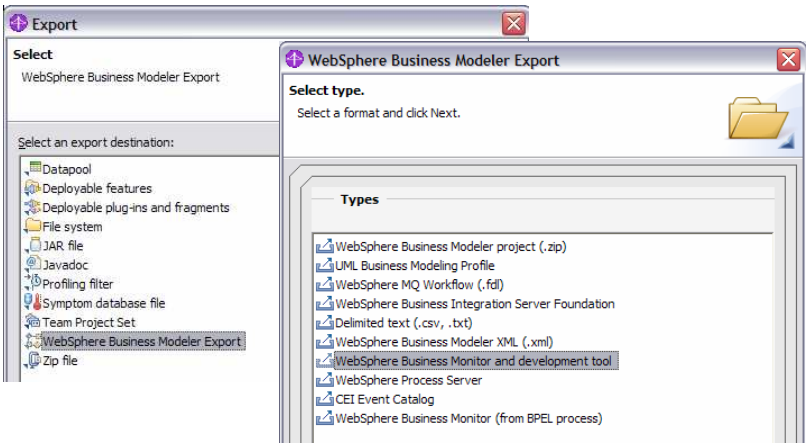
- Implementation may be set for tasks and services
  - Overrides Resource setting
- Empty implementation generated as a separate component during export
- Setting of [none] will generate implementation based on Resource setting
  - Setting of Machine generates Java component
  - Setting of Role or Person generates Human Task component
- Setting acts as a hint to assembler on what service to implement or what service to connect



The Implementation tab or values primarily applies to local/global tasks. The underlying implementation can be specified and an empty implementation of that particular type will be generated while exporting from Modeler and recognized as an empty component by WebSphere Integration Developer. If None (the default type) is specified for a particular task, it will generate an implementation based on a number of factors, including the type of Resource specified for that task. If a machine is specified, a Java component is generated. If a role or a person is specified, a human task component is generated. If the task is a local task and the resource is human, then the type of implementation becomes an inline human task inside the BPEL process. A local task with Java specified as the implementation will become a Java component inside the module. Because the implementations are empty even when a specific type is selected, it is best to think of the implementation type as a hint to the Integration Specialist (or whoever is performing the assembly of the process to services) as to what component should be connected for that activity in the BPEL process.

Export for WebSphere Integration Developer

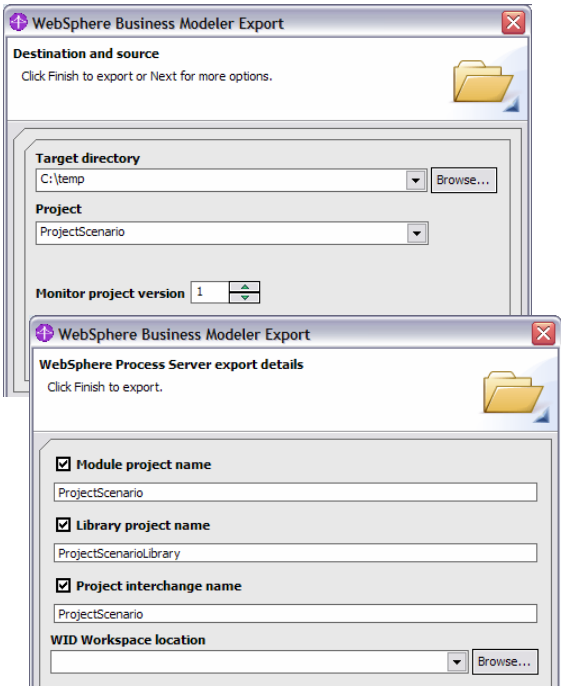
- WebSphere Business Monitor and development tool export option outputs artifacts in consumable format for WebSphere Integration Developer as well as any Business Measures
  - Other export options may result in a format consumable for WebSphere Integration Developer, but allow for business measures (Valid From Date) to become out of synch



With the process and tasks customized, the process is now ready for export. There are a number of export options that will result in a package that can be imported into WebSphere Integration Developer. While these will function properly, the best export option to use is the WebSphere Business Monitor and development tool option. This option will export the process and related artifacts as well as the business measures used by WebSphere Business Monitor for Monitoring the process. During this export, the process and business measures are synchronized with the same valid from date. This date is used by WebSphere Business Monitor when processing events. If the dates are not the same, WebSphere Business Monitor will ignore and discard the events. The other export options do not synchronize the date and leave open the opportunity for an incorrect value to be specified.

Export (cont.)

- Specify different Monitor project version number if a process and business measures have been modified
- Specify Module project name
  - Process and local
- Specify Library project name
  - Interface and Business Items stored in a library rather than module
- Specify Project interchange name
  - Results packaged in a ZIP file



With the WebSphere Business Monitor and development tool export option there are a few values to specify, including the destination directory and project. If you have made updates to an existing process and business measures, you should update the monitor project version value. This will allow for events from the new version to be differentiated from the events of an older version of the process. On the next screen of the export wizard you should specify the module name and a library name. If a library is not specified, all of the interface and business item information will be placed in the module, making it more difficult to use with other components. You should also specify a project interchange name, which will result in all of the contents being packaged in a single ZIP file rather than in a directory structure. You can also export the artifacts directly into a WebSphere Integration Developer workspace. However, this option is not recommended because it will overwrite any existing files, which could contain implementation details.

Agenda

- Overview
- Preparations in WebSphere Business Modeler
- Customizing in WebSphere Integration Developer
- Summary

This section will provide an overview of customizing the process in WebSphere Integration Developer.

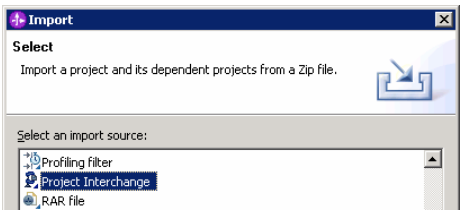
Preparing for Deployment – Overview of Steps

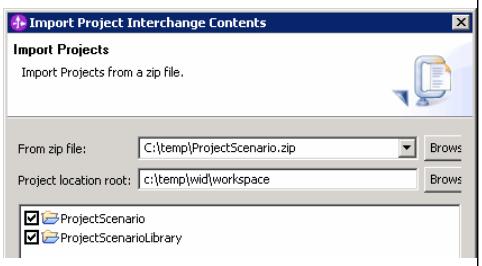
- Import process and artifacts
- Generate implementation for components or wire existing components
- Implement any snippets in BPEL process

Some primary steps are necessary for customizing the process once it is imported into WebSphere Integration Developer. After importing, you must complete the implementation for any components or connect to existing components. If there are any snippets in the BPEL process that are the result of using unsupported elements in the process in WebSphere Business Modeler, you must implement those as well. Each of these steps will be discussed in more detail.

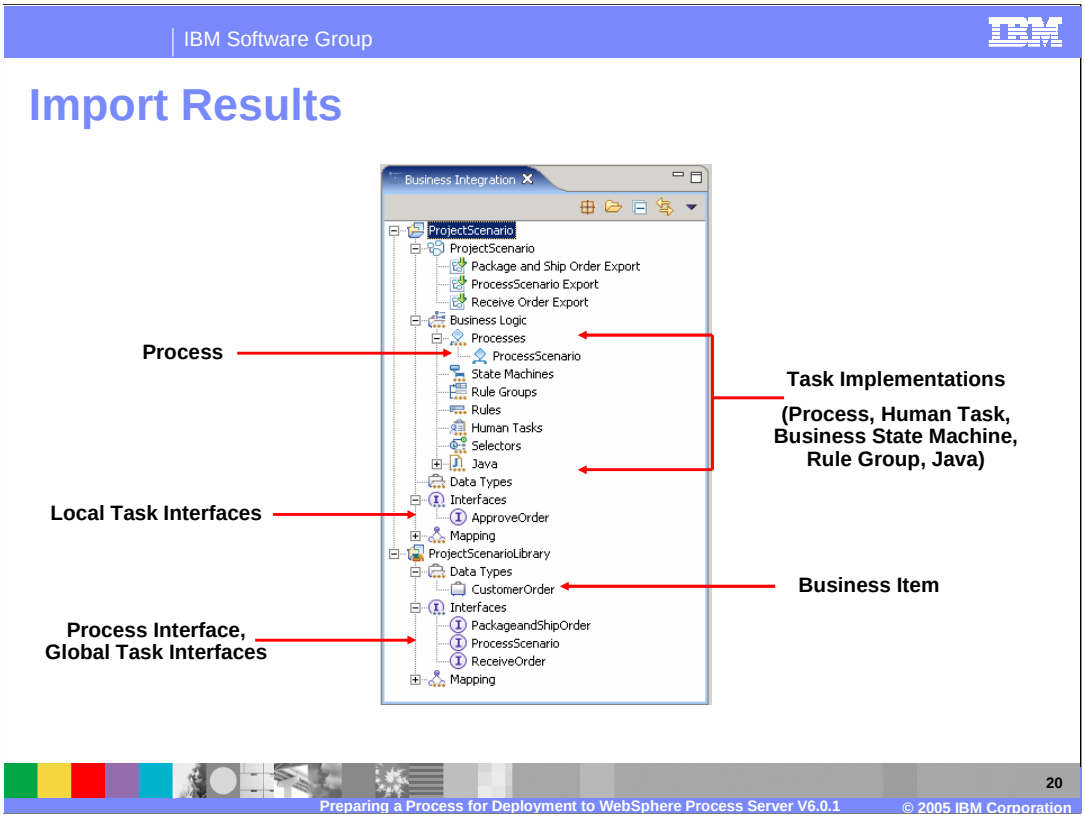
Importing Processes

- Processes and artifacts are imported as a Project Interchange
- All artifacts should be selected
  - Unselected artifacts will result in missing dependencies or out of synch resources
- Any existing artifacts modules or libraries with the same name will be completely deleted
  - Replaced with Project Interchange contents
- Partial Exports may be imported using Import > ZIP file
  - Not Recommended





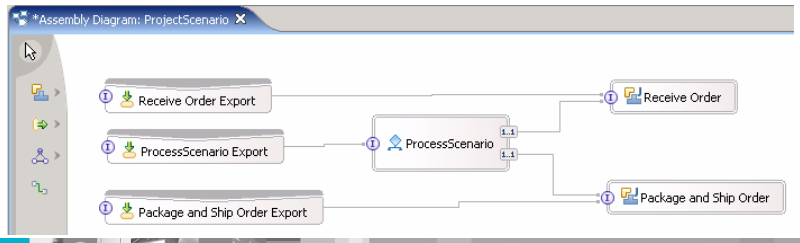
The Project Interchange import option should be used, rather than individual files, for importing artifacts into WebSphere Integration Developer. Any existing artifacts, modules or libraries with the same name will be completely removed and replaced with Project Interchange contents. If you are importing a process that you already implemented in that workspace, it is important that you back up the appropriate modules and libraries accordingly.



When you import a project into WebSphere Integration Developer V6.0.1, you will find that the process in WebSphere Business Modeler becomes a BPEL process. The task in the process will have empty components, corresponding to the implementation type value specified for the task. If any of the tasks are local tasks, the interface is created in the module. If any of the tasks are marked as global and a library was specified at export time, the interface will be placed in a library project. The interface for the process and the business items, in the form of Business Objects, will also be placed in the library if it was selected during export from WebSphere Business Modeler.

Implementing Tasks

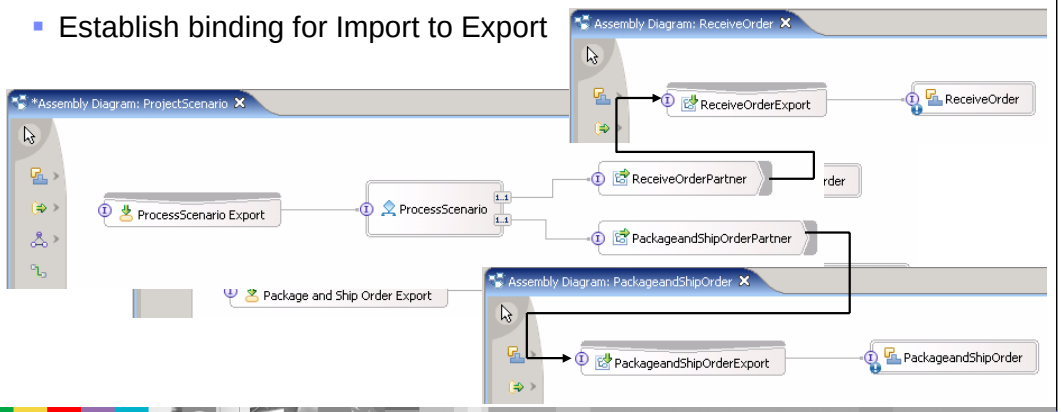
- Certain tasks are separate components in the same module as the process
  - Local tasks which have a human task implementation are inline human tasks in the process
- Tasks must be implemented
  - Implementing component in the same module advantages
    - Single deployable unit (EAR)
    - Better performance between components
  - Implementing component in the same module disadvantages
    - Establishes a tight component model which is less flexible
    - More difficult to maintain on process updates



As previously indicated, the various tasks that comprise the process will be implemented as empty components of a particular type. These tasks can be opened and implemented from the Business Integration view or from the Assembly Editor inside WebSphere Integration Developer. When implementing the tasks, you have a variety of options. If a service exists, the empty component can be removed and the process reference can be wired to reuse that existing service. Existing services are typically in a separate module and you would use an import component to represent the service. If the service does not exist, you can implement the empty component in the same module. There are advantages and disadvantages to implementing the component in the same module. One advantage is that performance between components within the same module is better than components in separate modules. Another advantage is that there will be fewer applications to manage in the production environment, because each module maps to an enterprise application or EAR file. A disadvantage of implementing the component in the same module is that if there is a need to change one of the components, whether the process or a component based on a task, the entire module with both components must be updated in the production environment. Also, with the components directly connected, there is no way to change the binding information. With the components wired in separate modules, wiring can be changed in production, making it possible to use a different component with the same interface. Everything in one module also makes it difficult to maintain updates when a new version of the process is assembled in WebSphere Integration Developer and deployed to WebSphere Process Server. For most environments, the disadvantages outweigh the advantages and it is better to use separate modules rather than have all of the components in the same module.

Implementing Tasks in Different Module

- Replace empty components with Imports
- Implement tasks in separate modules
  - Expose with Export
  - Use component type as “hints” to what type of service component should be used
- Establish binding for Import to Export

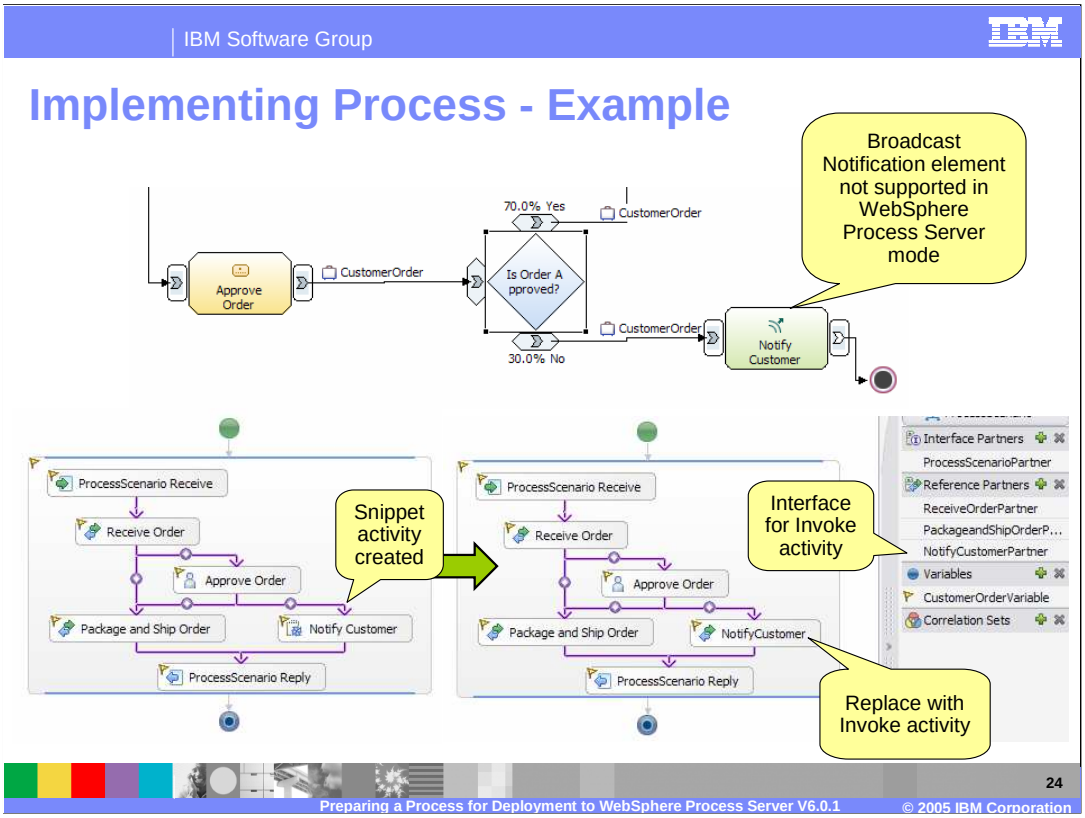


This example further emphasizes the advantages of using separate modules for task implementations. By default, the process has empty components generated in the same module as the BPEL process. These components can act as hints to indicate what type of component should be used in another module. The component can be replaced with an Import and a binding to the Export of the component in another module can be made. Using this solution, a change to the process requires only a set of rewire operations by the integration developer, and not a change to the components.

Implementing Processes

- Certain tasks and constructs from Modeler do not have matching BPEL activities
  - Implemented as empty Snippets
- Snippets can be replaced with Invoke activities or custom library snippets can be created and reused
  - Interfaces for Invoke activities should be placed in a separate library
- Certain BPEL concepts and constructs do not have Modeler equivalent and should be used carefully
  - Faults, Fault Handlers, Compensate, Compensation Handlers, Wait activity, Scopes (not associated with a local sub-process in WebSphere Business Modeler), Expirations, Event Handler

In addition to the various tasks, there is implementation that must to be done in the process. Certain tasks and constructs that are used in WebSphere Business Modeler are not supported in BPEL and will be implemented as empty snippets, which need the appropriate code added. Any code that you add will be lost if you import a new version of the process. You can make use of the custom snippet support in WebSphere Integration Developer and then reuse the code when you make updates to the process. You can also replace snippets with invoke activities, which call out of the process to components existing in the same module or, preferably, in a different module. When you make future updates to a process, the snippet would just need to be replaced with an invoke activity again. Certain BPEL concepts and constructs, such as faults, fault handlers, compensate, compensation handlers, wait activity, scopes that are not associated with local sub-processes, expirations, and event handlers do not have equivalent elements in WebSphere Business Modeler and should be used carefully by creating them in a separate process in a separate component. You should make as few changes to the BPEL process as possible.



This slide shows an example of implementing a process containing snippets. In this process, if an order is not approved, the customer must be notified. The Notify customer task is a notification, which is not supported when exported to BPEL. A snippet is created in the process and must be implemented to perform the notify customer option. Putting in your own code will result in a loss of that code if a new version of the process is imported into WebSphere Integration Developer. Therefore, it is better to replace the snippet with an invoke activity and implement the logic in a component in a separate module. An interface, defined in library, will be associated with that invoke activity through a partner defined on the process.

Other Considerations and Notes

- Do not change interfaces or business objects in WebSphere Integration Developer
  - Use Interface and Data Maps if existing implementation interfaces and data objects are different
- Limit amount of changes to BPEL process
  - Use Business Rules, custom Snippets, and Invoke activities to other components for additional logic and processing
- Do not modify Event Monitoring settings in BPEL process
  - Settings defined in Modeler for monitoring by WebSphere Business Monitor
- Do not change the Valid From date setting
  - Changes will cause generated events to be ignored and discarded by WebSphere Business Monitor

There are some important things to consider when assembling a process in WebSphere Integration Developer. It is best not to make any changes to interfaces or business objects in WebSphere Integration Developer. If you are connecting the process to existing services, use interface maps and data maps if the existing service interfaces are different. Limit the number of changes to BPEL process. Any changes you make will be lost when you import a new version of the process into WebSphere Integration Developer. WebSphere Business Modeler will set event monitoring settings for the BPEL process, variables, and Invoke activities. These settings should not be changed because it could affect how WebSphere Business Monitor processes and records events for the process and reports business metrics and key performance indicators. Finally, do not change the Valid From date setting in WebSphere Integration Developer. This value is set at the time the process is exported from WebSphere Business Modeler and will match the Valid From Date on the business measures. Any changes to the date could cause WebSphere Business Modeler to ignore and discard events generated from the process.

Agenda

- Overview
- Preparations in WebSphere Business Modeler
- Customizing in WebSphere Integration Developer
- Summary

This section will provide a summary of this presentation.

Summary

- WebSphere Business Modeler allows for customization of technical attributes for processes and nodes
- Implement components and services with WebSphere Integration Developer

In summary, WebSphere Business Modeler allows for technical attributes to be specified for processes and nodes prior to export. With WebSphere Integration Developer, processes can be further implemented in preparation for deployment to WebSphere Process Server.

New BPEL/WSDL Generation Support

- BPEL <pick> activity generation
  - By defining multiple Input Criteria on the Process, which represents multiple exclusive trigger points of the Process, Modeler will generate a BPEL <pick> activity
- BPEL <switch> activity generation
  - An option is provided that is presented as a Technical Attribute on a Decision node to generate a BPEL <switch> activity for the Decision node
- Support of one-way operation
  - Support the configuration of the WSDL operation type through Technical Attributes
  - User can specify whether the WSDL operation to be generated is a one-way or request/response type operation
- Optimized the number of variables generated
  - In Modeler 5.1.x, the BPEL variables are not re-used between the input and output of a BPEL activity, thus many variables are often created. In 6.0, the number of variable generated has been optimized and reuse of the variables is done as much as possible
- Optimized transformation of control actions (Fork, Decision, Merge, Join)
  - In Modeler 5.1.x, control actions like Fork, Decision, Merge, and Join may generate a BPEL <empty> activity
  - In 6.0, the transformation is optimized and will no longer generate unnecessary empty activity
- Generation of WS-I compliant WSDL Message
  - In 6.0, the WSDL Message generated will be fully conform with the WebSphere Integration Developer convention using a document-literal wrapped style
- Technology Mode support
  - In 6.0, the concept of User Profile and Mode in 5.1.x has been unified into a single concept of Mode
  - There will be 6 different modes including Basic, Intermediate, Advanced, WebSphere MQ Work Flow, WebSphere Business Integration Server Foundation, and WebSphere Process Server
  - The later 3 technology related modes will sit on top of the advanced mode which exposes all the attributes in the model
  - Technical Attributes support will be available only in the WebSphere Process Server mode.

Trademarks, Copyrights, and Disclaimers

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

IBM	CICS	IMS	MQSeries	Tivoli
IBM(logo)	Cloudscape	Informix	OS/390	WebSphere
e(logo)business	DB2	iSeries	OS/400	xSeries
AIX	DB2 Universal Database	Lotus	pSeries	zSeries

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

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

Intel, ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds.

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

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or program(s) described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (e.g., IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

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

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.