•There are some terms and concepts which are common to many business processes and workflows. It is important to define these before looking at a specific business process runtime environment.
In this presentation, we will look at why Process Choreography is needed in today’s businesses and define and discuss the key terms around the concept. We will also look at some of the key concepts around executing a Business Process using the newly emerging language standard for describing business process flows, BPEL.
Business Process Choreography (BPC)

- A set of functions that allow the composition, arrangement, execution, and administration of business processes
- A business process is a set of logically connected services
  - Each service is a black-box functional unit
- A business process typically defines:
  - the rules that govern how and when the services are called (sequences and flows)
  - the interactions with people (staff) and the environment (events)
  - the actions to be taken to "undo" the effects of the execution (compensation)

A Web Service

• The Business Process Choreographer (which implements the Business Process Choreography) is a set of tools and functions that allow you to arrange a number of activities (individual functions) in a flow. Most concepts that apply to the BP Choreography also apply to workflow technologies.

• Each business process is a set of logically interconnected services. A service is a black box functional unit which is wrapped and represented by a WSDL (Web Services Description Language) file, as defined by the Service Oriented Architecture.

• A business process defines the flow in which these services are invoked - their sequence and any conditional processing. It also defines the modalities for human intervention and how external events, possibly coming from other computing systems, have to be handled.

• A special aspect of business processes is that they may be long running processes, made of a number of individual transactions. Traditional "rollback" operations do not apply to business processes, because a business process may span multiple physical transactions - and also involve non-transactional operations, such as sending a confirmation via e-mail or printing a form. If a business process needs to be "rolled back", the developer has to define appropriate actions to "compensate" those activities that were already completed. For instance, in order to compensate a confirmation e-mail, the business process might send a cancellation e-mail.
What is a Service-Oriented Architecture?

- An approach to building distributed systems that delivers application functionality as services to end-user applications or to other services.

- A flexible and open architecture that…
  - represents software assets as services
  - has well defined ways of representing and interacting with software components
  - can integrate with applications inside and outside of the enterprise

- Software components become reusable building blocks
  - Focus on application assembly rather than development
  - Create new internal apps out of existing components
In a perfect world, businesses would be able to …

- describe a process and model the process graphically
- pull the services needed from a palette into the process map
- drag and drop relationships between data, people, systems and services
- identify and mark measurement points
- simulate the operations for a month and make changes
- customize the solution with deployment options
- test your process and make sure that it runs
- collect, analyze and compare operational performance against the simulation
Business Process Execution Language (BPEL) specification
BPEL Overview

- Business Process Execution Language for Web Services (BPEL4WS, BPEL) provides a standard for defining the business process model

- Based on WSDL and other XML standards
  - WSDL defines the interface of services used by BPEL
  - XML Schema and XPath for data context handling and business rules specification
History and Design of BPEL

- BPEL v1.1 is the current specification
  - 136 page document finalized in May 2003
  - Supported by IBM, BEA, Microsoft, SAP, and Siebel
  - Is an update to BPEL v1.0 finalized in August 2002

- BPEL defines a clear separation between the business process model, the implementation of activities, and runtime execution

- External partners (services) are invoked as Web Services and the business process becomes a Web Service too

• The BPEL specification and its concepts have been around for quite a while. Originally started by IBM, BEA, and Microsoft, the specification has become more refined and is supported by SAP and Siebel now as well.

• BPEL defines 3 separate parts or phases of a business process architecture.
  • The development of a business process model
  • The implementation of the activities that are involved in the business process
  • The runtime engine that executes the business process model by determining the correct path through the business process and calling the necessary activities.

• External partners are invoked as Web Services

• The BPEL business process should also be available as a web service so it could potentially become an activity or sub-process of a larger process
This is an agenda of sorts. Going through all these types elements and learning why they are important in BPEL.
Main Elements of a BPEL Business Process

- Partner Links
  - Declared service providers for the implementation of the activities
  - Linked to specific activities through Port Type
- Variables
  - Hold data that constitute the state of a process
- Correlation Sets
  - Uniquely identify a business process instance through a set of known variables
- Fault Handlers
  - Handle “planned” errors or exceptions within the business process
- Activities
  - Basic activities
    - A step in a business process like receiving input, making decisions, managing exceptions, and replying to partners or other business processes
  - Structured activities
    - Organize and manage the complexity of the flow, similar to programming constructs

A business process model is composed of the different activities which make up the business process as well as the order in which these activities should be called.

The order and control of the flow is provided by the structuring activities.
BPEL BASIC Activities

- **Receive**: Wait for a message to arrive. Optionally start a new process instance when the message arrives.
- **Reply**: Send a message in reply to a message that was received through a Receive.
- **Invoke**: Invoke a one-way or a request-response operation offered by a partner.
- **Assign**: Update the values of variables with new data.
- **Throw**: Generate a fault from inside the business process.
- **Empty**: A "no-op" instruction in the business process.
- **Wait**: Wait for a given time period or until a certain time has passed.
- **Terminate**: Immediately terminate the process instance.

**Receive** is another asynchronous activity which can be added to a business process. When a Receive activity is reached, the business process will halt and wait for a specific operation to be called from an outside location, in theory, synchronizing the business process with an outside action. When the operation is called, the business process will continue. The Receive activity is a blocking wait.

The **Empty** activity acts a placeholder in a business process for a future activity which may be implemented. When the Empty activity is reached, the business continues without stopping, treating the activity as a no op. The Empty activity can be morphed at a later time to any of the other activity types.

Business processes can also **Invoke** other business processes or Web Services during execution. Using the Invoke activity, you can reuse existing business processes within a parent business process. If the top-level business process is executed as an interruptible business process, the called business process can be interruptible or non-interruptible. If the top-level business process is executed as a non-interruptible business process the called business process can only be non-interruptible.

The notion of sub-processes is an IBM BPEL extension and the behavior is controlled by the ‘autonomy’ attribute of the process.

Business processes can fail at a business process level, which is something that is acceptable and can be part of the business process model. When a Fault occurs in a business process, the **Fault Handler** is invoked and the fault can be managed at the current scope or **thrown** to higher level scopes, signaling the business process has failed, or the business process can **terminate**.
**BPEL Structured Activities / Elements**

- **Sequence**: Holds multiple activities that are performed sequentially in lexical order.
- **Flow**: Holds multiple activities that are performed concurrently.
- **Switch**: Selects one branch of activity from a set of choices.
- **Link**: Synchronizes two activities that are enclosed in a Flow (i.e., enforces a certain execution order).
- **While**: Holds a (basic or structuring) activity that is repeated until a success criteria has been met.
- **Pick**: Wait for one of multiple messages to arrive or for a time-out alarm to go off.
- **Scope**: Allows for the definition of a nested activity with its own fault handler.
IBM Extensions of BPEL: New Activities

- **Staff**: Invoke an interaction with a human user.

- **JavaSnippet**: Invoke an inline snippet of Java code.

- **Subprocess**: Invoke another process – life cycle events will be propagated.

**Staff** - The Human Interaction activity is a point in the business process where a person must act and must complete the activity for the business process to continue. The Human Interaction activity is an asynchronous activity as there is no guaranteed time in which the person will actually complete the activity. A business process with a Human Interaction activity should be executed as an interruptible business process to insure that the current state of the business process has been saved while waiting for the Human Interaction activity to complete.

**Java Snippet**: a piece of inline Java Code

**Subprocess** – Invoke another business process as if it were an activity in the parent business process. Life cycle events can be propagated between the processes.
Products and Tooling Support
In the area of Business Process Choreography, WebSphere Studio IE v5.1 includes specific tooling to allow developers to graphically build business processes by dragging-and-dropping activities and interconnecting them using the appropriate links. Business Processes can also be tested directly within the tool.

The tool provides special interfaces towards user registries (operating system or LDAP) to enable staff support and human interaction. It also facilitates the construction of compensation pairs and it includes a graphical debugger that allows you to step through a business process.

The WebSphere BI Server Foundation v5.1 runtime supports installing and running applications that implement business processes. The runtime implements what is known as the "Business Process Engine", which tracks the progression of each instance of a business process. Administrative support is provided through the administrative console to allow the configuration of the Business Process engine and the administration of the business process templates.

Long running (interruptible) business processes are fully supported - the state associated with such processes is stored in a relational database.
The Tooling Perspective

WebSphere Studio family and WebSphere Application Server family

A next generation integrated development environment for build to integrate solutions

WebSphere Studio
Enterprise Developer

WebSphere Studio
Application Developer Integration Edition

WebSphere Studio
Application Developer

WebSphere Studio
Site Developer

WebSphere Application Server

WebSphere Application Server - Express

WebSphere Business Integration Server Foundation (WBI SF)

WBI SF for z/OS

The core J2EE technology and Web services deployment environment
Business Integration Perspective

Outline View

Business Process Editor

Services View
The Business Process Editor – Terms and Layout

Business Process

The Canvas

The Palette

Details Pane

Minimize, Maximize, and Restore Details Pane
WebSphere Business Integration Server Foundation v5.1

- WebSphere Application Server Enterprise re-branding
  - WebSphere Business Integration Server Foundation
  - Base for higher value WebSphere Business Integration solutions
  - Prelude to tighter integration with more specific BI products
  - Used as the runtime environment for building composite applications
    - Same audience as for WebSphere Application Server Enterprise

- Based on WebSphere Application Server v5.1

- Focused on integration
  - Maintain and enhance process choreography
  - Add new integration capabilities and easier to use tooling

- Available to z/OS customers
WebSphere Business Integration Server Foundation v5.1 Themes

- Provide application enablements beyond the J2EE standards
- Enable new classes of applications to run on WebSphere
- Three areas of new features and improvements:

**Business Object Model**
- EJB 2.0 (CMP, CMR, QL...)
- Caching mechanisms
- Application profiling
- Dynamic Query Service
- Read-ahead, prefetch...
- Extended EJB lifecycle
- Advanced CMP

**Business Process Model**
- Business Process Choreographer using BPEL4WS
- Activity Sessions and Last Participant Support
- Business Rules Beans (*)

**Next Generation Apps**
- Extended Messaging Support
- Asynchronous Beans
- Scheduler/Calendars
- Startup Beans
- Object Pooling
- i18N (*)
- WorkArea (*)

**Base WBI SF**
- Web Services Support
  - JSR 101 and 109
  - Services Oriented Architecture
  - Web Services Gateway & UDDI

**Base**
- EJB 2.0 (CMP, CMR, QL...)
- Caching mechanisms

WebSphere Business Integration Server Foundation V5.1 includes a wide and comprehensive set of new enablements for the advanced application developer.

This chart groups those enablements in three different areas.

A first group of functions targets the business object model, that is, the representation of the business data by a network of Entity EJBs. In this area, WebSphere BI Server Foundation v5.1 provides a series of extensions to the standard EJB 2.0 specifications and to the data access and caching functions supported by the base WebSphere Version 5. In this category you can find functions such as Application Profiling, to optimize data access through CMP EJBs. Dynamic Query Service allows you to formulate dynamic EJB QL queries that can be submitted at runtime, as opposed to the static EJB QL support mandated by the specs. This layer also provides you with the dials and knobs that allow you to fine tune the caching and pre-fetching mechanisms of the application server - and to influence the EJB lifecycle, by preventing unwanted passivation of the entity EJBs.

The second group of functions is targeted to the business logic developer. Here WebSphere BI Server Foundation v5.1 introduces one very significant new technology: the Business Process Choreography. This technology allows developers to organize business activity in a potentially long-running workflow, called a business process. The tooling provided in this area allows the visual composition and debugging of such flows, while the runtime supports the execution and administration of business processes. Still in this area, the Activity Session service allows you to extend the semantics of a global transaction to involve single phase resources. Single phase resources cannot, by definition, participate to unit of works where multiple resource managers are involved. The Activity Session service removes this limitation. Also in this area, WebSphere BI Server Foundation v5.1 continues support for the Business Rules Beans, as a carry over from Version 4. BRBeans allow the development of rule base applications.

The third family of functions provided by WebSphere BI Server Foundation v5.1 provides advanced application enablements - to greatly facilitate the development of sophisticated applications. Extended Messaging Support (EMS), for instance, makes it faster and easier to interface J2EE applications with messaging infrastructures. Asynchronous beans enable true multithreaded programming in the EJB container - the Scheduler allows creating tasks that can be executed in unattended mode and repeated at regular intervals.
Supporting Business Processes

- **WebSphere Business Integration Server Foundation v5.1 supports:**
  - Running business processes based on BPEL (with extensions)
  - Running business processes created with v5.0 technologies
  - Synchronous and Asynchronous business process interfaces
  - Maintaining state of long-running business processes
  - Activity expiration using calendars in business processes
  - Assigning Work Items to individuals who can participate in the business process
  - Managing and monitoring business processes via the Web Client
  - Options for running business processes in high availability environments

- **This support is available to business processes through the Business Process Container in WBI SF v5.1**

- There are many existing IBM products which provide workflow technology for many different situations, scenarios, and implementations. While IBM is currently offering this technology in different products such as WebSphere MQ Workflow for people-based workflows, WebSphere MQ Integrator for message flows, WebSphere MQ Adapter Offering for MQ-based adapters, and Enterprise Access Builder for Java-based adapters, no tight integration of workflow technology into WebSphere is currently available. WebSphere Business Integration Server Foundation brings flow support into the application server, allowing customers to easily combine workflow technology with all the other services offered by J2EE, tightly integrated within WebSphere. With this tight integration, the many features of WebSphere Application Server, such as centralized management, workload-management, logging, debug and trace, can be used when executing business processes. Besides workflow enabling J2EE application and components, the intent is to allow for independent Web Services to be coordinated along with J2EE applications. The support has been designed in such a way that compliance with the BPEL (Business Process Execution Language) industry standard can easily be met in the future. The workflow support will allow for many new integration opportunities for developers.

- The workflow support is a pure Java implementation which runs on the application server. It has been designed with a flexible and extendable design allowing for support for many different technologies while executing business processes.
Business Process Container
Business Process Container - Implementation Resources

- The Business Process Container is an Enterprise Application
  - bpecontainer.jar - External Interfaces of Business Process Container
    - Synchronous Session bean and Asynchronous Message-driven bean interface to Business Process Container
  - compensate_ejb.jar
    - Compensation beans

- It requires:
  - A relational database and a data source
    - CloudScape (test), DB2, Oracle, Sybase, DB2 for z/OS
  - A JMS provider and a number of JMS resources
    - WebSphere embedded messaging, WebSphere MQ, ...
Business Process Container - 4 Message Queues

- Queue for Asynchronous requests into the Business Process Container
  - Used by asynchronous interface for external client requests
  - Name: BPEAPIMQueue

- Internal Queue for processing interruptible Business Process activities
  - Used for executing the activities of an interruptible Business Process
  - Name: BPEIntQueue

- Retention Queue for retrying messages (activities)
  - Used for retrying activities which fail
  - Name: BPERetQueue

- Hold Queue for failed messages which will not be retried
  - Used for holding messages which fail and have exceeded retry limit
  - Name: BPEHldQueue

- There are actually four message queues used by the Business Process Container. The first queue, BPEAPIMQueue, is used to support the asynchronous interface into the Business Process Container. A message-driven bean with the help of a listener port monitors the destination for messages to start Business Processes or call events within a Business Process instance. When the message comes into the message-driven bean, the request is processed by the Business Process Container.

- The other three queues are part of the internal Business Process Container execution of different Business Process activities. The Navigator will place the next activity of an interruptible Business Process into the BPEIntQueue destination. A message-driven bean is monitoring this destination and will pull the message off for processing. The message is passed to the Navigator which will match the message to an activity in a particular Business Process instance. The activity will be executed and the result will be stored within the BPEDB along with a record that the activity completed successfully. A message is then placed in the BPEIntQueue destination again indicating the next activity to be called for the instance and the execution repeats until all the activities in the interruptible Business Process are completed. This destination is also used for receiving response to asynchronous invocations.

- The BPERetQueue destination is used when a message from the BPEIntQueue destination results in a failure in the called activity. The message will be placed in this destination where it can be retried a certain number of times. Moving the message to a different destination allows processing to continue for executing other valid messages and activities. The number of times the message is retried can be configured in the Business Process Container in the Admin Console.

- The final destination, BPEHldQueue, is used when a retried message continues to fail. The message is placed within this destination allowing failed messages to be retried without forcing the listener port to stop listening to the BPERetQueue destination. By default no message-driven bean monitors this destination, however, an administrator can force messages to be processed from this destination and retried.
### Business Process Container - Building Blocks

**Components**
- Process Navigation - Coordinates the calls between different activities and calls to external plugins
- Factory - Manages the state information of a Business Process with persistent store
- People Interaction - Creates and maintains Work Items for users for different activities within a Business Process

**External APIs**
- Synchronous session bean interface for starting and working with business processes and work items
- Asynchronous message-driven bean interface for starting business processes
- Specific session beans – used with BPEL processes

---

- Included with WebSphere Business Integration Server Foundation v5.1 is a Business Process Container which provides the support for Process Choreography. The Business Process Container is made up of components, plug-ins, and APIs.
- The Components (Navigator, Factory, and Work Item Manager) are the main items responsible for the execution of business processes. The Navigator does much of the work with executing the different business process instances. The Navigator refers to the business process template and executes the correct activity in sequence. The Navigator uses the Factory component when executing long-running interruptible business processes. The Factory is responsible for storing the state information of interruptible business processes. After each activity is completed, the state is stored from where it can be read when the next activity is to be executed. When executing the next activity in an interruptible business process, the Navigator will call the Factory for the specific instance state which has been stored and it will be returned to the Navigator. The Navigator will then use the information to call the next activity. When the activity completes, the result will be returned to the Factory for storage. The third component is the Work Item Manager. When the Navigator reaches a Staff activity, the Work Item Manager is called and is responsible for creating Work Items for users with different permission levels.
- The Plug-ins which are included in the Business Process Container help the three main components. The Data Provider plug-in allows the Navigator to handle different message types sent to and received from activities. By default, the Data Provider plug-ins included are Java and XML. These plug-ins allow the messages sent to and received from the activities to be Java objects or XML described messages. The Invocation Provider included with the Business Process Container is based on a service-oriented architecture provided by WSDL (Web Service Description Language) and WSIF (Web Service Invocation Framework). Each activity in a business process is called in this manner. The next plug-in, Staff Plug-in Provider, is used by the Work Item Manager for searching specific user registries. The final plug-in is the State Observer which interacts with the Factory and provides status information of what activities the different business processes have completed.
- The final important part of the Business Process Container is the set of APIs. The APIs allow for external clients to start business processes, send event messages, and view the current state of the business processes. There are also APIs which allow clients to Claim and Complete work activities.
- With the separation between the components and the plugins, the Business Process Engine can focus on executing processes and not supporting other services. Also, different plugins specific to a particular environment can be easily added in the future.
- Accommodates different types of clients
- Modular, extensible architecture (extensible for e.g. Staff)

This diagram shows the Components, Plug-ins, and APIs which make up the Business Process Container described in the last slide. Notice the Factory and the Audit Trail Observer Plug-in utilize a database for storing the state information of a business process or for finding the state of business processes.

The external APIs for starting a business process or for sending events are exposed over IIOP or JMS. There is a session bean and a message-driven bean which clients can call for sending messages into the business process database. The session bean provides a synchronous connection for a client and the message-driven bean provides an asynchronous interface.

The asynchronous and synchronous facades provided by the message-driven bean and the session bean are part of the Business Process Container. WebSphere Studio Application Developer Integration Edition actually generates facades which call the session bean part of the Business Process Container.
Basic Business Process Concepts
Business Process Templates and Instances

- **Process templates...**
  - contain all information about a business process model
  - are used to create process instances at run time
  - are deployed and installed via the Admin Console
  - can be started and stopped via the Admin Console

- **Process instances...**
  - are individually created from a process template
  - from same process template can be executed in parallel
  - can be started, stopped, and monitored in many ways
Business Processes

- Two major types of Business Process Execution

  - **Non-Interruptible** processes are short-lived operations combined to form a single business operation
    - Example: Validate credit card information

  - **Interruptible** processes are long-running and are composed of business services
    - Example: Loan approval
    - Example: Document review

There are two major types of business processes which can be executed. The first type is non-interruptible which is used for executing short business processes or possibly a small unit of work composed of multiple steps within a larger business process. With a non-interruptible business process, it will either complete or fail. No intermediate state can be maintained. Consider a credit card verification process. This business process is very short and simple. accepting an individual’s information and returning a result which indicates if the individual’s credit card was valid. In this business process, it may start with a number of lookups which check information about the individual before returning the results. These lookups will either succeed and return a result or fail and the entire credit card validation will fail. The failure could be monitored by the client who could call the business process again. In any case, no state information is being maintained from the different lookup steps.

The second type of business process is interruptible. These business processes are long-running and may take hours, days, or weeks to complete, and because of the possible random completion time, state must be maintained during execution. If state was not maintained and the current execution thread running the business process would end (for example when the server is restarted), the work which would have been accomplished would have been lost. An example of an interruptible business process would be a loan approval process. With a loan approval process, it will start by gathering an individual’s appropriate financial information and then evaluating this information. If the individual appears to be financially sound, they can be offered a loan and at the same time the funds may be reserved. Since the evaluation of the financial information may take a number of days, the state of this business process will need to be maintained. After each activity in the process is completed, the results should be recorded to keep track of which information was processed and how much was completed in the loan approval process.

In WSAD IE 5.1 there is an option to mark a given process as long-running, to indicate that it is interruptible, or not.

The terms microflow and macroflow are v4.1 terms and have been deprecated.
Non-Interruptible Business Processes

- Short running
- Encompass a single transaction
- Run in a single thread
- Synchronous execution
- Client can wait until business process completes
- No persistent state is maintained during execution

• Since a transaction is associated with a single thread of execution, there can not be parallel processing in a non-interruptible business process. Even if the business process model has defined multiple activities to be executed at the same time, the actually execution environment will execute one activity and then the other with no particular order.

• Because of the single thread of processing, a non-interruptible business process can be invoked with a synchronous call. When the business process completes, the results can be returned directly on the interface which was used to call or start the business process. For most non-interruptible business processes, the client can call the business process and wait for the result.

• Remember, no state is maintained during the execution of the business process. All the results and state is contained in the thread of execution. Should that thread end prematurely (server shutdown), all the progress of the business process will be lost and the business process would have to be executed from the beginning.
Transactions: Non-Interruptible processes

Non-Interruptible

Receive
Invoke
Invoke
Invoke
Reply
Reply
Fault

Transaction boundary

T₀
Executing a Non-Interruptible Business Process

- Client request starts a Business Process instance through external API
- Client Transaction and Security context used
- Navigator executes all activities sequentially
  - Single thread, single transaction
  - Database (Factory) and JMS destinations not used
- Successful completion of the process commits the transaction

• When a non-interruptible business process is executed by the Business Process Container, a transaction is started to execute the activities. The transaction is started with a TX_REQUIRED attribute. Any transaction or security context established by the client is used by the Navigator as it executes each activity in the business process. When all activities have been executed and the business process ends, the transaction is committed, or it is left to the client who established the transaction to commit the transaction.

• If there are multiple two phase commit activities within the Business Process and a single one phase commit activity, Last Participant Support is used to coordinate the execution, placing the one phase commit activity as the last activity to be committed. Should the one phase commit activity fail, the instance will fail and all the two phase commit activities will be rolled back.

• If a non-interruptible business process is started from an asynchronous interface such as through the JMS facade, the user id/password used to execute the activities is the one specified for authentication of the JMS destination.

• The different activities within a non-interruptible business process should contain an implementation which supports transactions. EJB implementations make the most sense for the different activities. SOAP or asynchronous activities are not appropriate as they do not have transaction capabilities and do not have built in support for rolling back a completed action. SOAP and asynchronous activities should only be used if inconsistencies are acceptable when a non-interruptible business process fails.
Interruptible Business Processes

- Long running
- Multiple Transactions
- Multiprocessing of different operations at the same time
- Execution can be paused
  - Wait for an outside event
- Completely forward recoverable
  - A Business Process which is suspended can continue execution from the suspended state
- Asynchronous execution
  - Client starts Business Process and continues execution
  - When Business Process completes, results can be passed to specific location
- State maintained during execution
  - State information stored between activities

Interruptible business processes are very different from non-interruptible business processes. These business processes are long running and may involve multiple transactions and fully support parallel processing if two activities in a business process have been defined to run at the same time. Each activity actually runs in its own transaction and it is the runtime environment’s responsibility to manage the execution and management of the different transactions to coordinate their execution and use the results of one transaction in other transactions (activities) later in the business process.

Interruptible business processes are executed and managed in such a way that their execution can be paused administratively or by the business process definition in the model. If there is an asynchronous activity, which may complete in an undetermined time, the business process can pause or wait until that activity completes. While waiting, the state of the business process at that point is stored, allowing the thread of execution to end and a new thread to begin with the stored information when the activity is resumed. This functionality is referred to as forward recoverable.

An interruptible business process may take hours, weeks, or even months to complete. Because of the uncertainty about when the business process will complete, a client may not be able to wait for completion on the same thread of execution used to start the business process. Some other technique must be used to notify the client that the business process has completed. In the interruptible business process, separate operations are defined which are called when the business process completes. These operations can be a client service which is called and can be passed the result information. In many situations, if the client does not have an active service available to be called and passed the result, the result can be passed to a queue. The queue will hold the result and the client can pick it up at a later time.

The state of the business process is stored after each activity has been completed, keeping track of the current position in the business process and also allowing for the forward recoverable capabilities.
Executing an Interruptible Business Process

- Request starts a single Business Process instance through external API
- Navigator executes each activity under individual transaction
  1. Transaction begins
  2. Activity is read from internal JMS destination
  3. Instance restored from Database (Factory)
  4. Activity Invoked
  5. Response stored in Database (Factory)
  6. Next Activity message place in internal JMS destination
  7. Transaction committed
- Security information is stored in a Database when process starts and at end of each call and reused when next activity is executed
- Failed activity may be handled and Business Process "rollbacks" can be performed

For interruptible business processes, execution is quite different from the single transaction behavior of a non-interruptible business process. In an interruptible business process the next activity to be executed is placed on a queue where it can be processed when the server resources are available. Since the interruptible business process is long-running, this is an acceptable implementation. A transaction is started by the Business Process Container as the message is read from the queue by the Navigator. The Navigator will reference the specific business process for the message and call the Factory. The Factory will return the state of the business process to the Navigator where it will be used to call the next activity. When the activity is completed, the result will be sent to the Factory for storage. The Navigator will then determine the next activity and place a message in the queue. The transaction will then be committed. The message will be read from the queue under another transaction and the next activity will be executed in a similar manner.

Execution of a business process actually uses the Asynchronous bean technology to execute the different activities under the client's security context, if an interruptible business process is started through the session bean API façade. If an interruptible business process is started from the asynchronous message-driven bean API façade, the user id/password used is the one specified for authentication of the queue. Using the Asynch Beans technology, the Security context is stored in the database between each call and reused when the next activity is executed. Asynch beans have a feature which allows work objects to store the associated security context and rebuild and reuse it later.

Since an interruptible Business Process may last hours, days, months, there is a chance for the security information to expire. Should the security information expire, a work item will be generated for the Business Process administrator and the administrator can continue the instance under the administrator's user id/password. Expiration of security information is considered a runtime error and cannot be handled within the business process.
Interruptible Business Processes and Transactions

Transaction

JMS message “Continue Connector”

JMS message “Activity Invoke”

JMS message “Activity Complete”
Controlling Business Process Execution
Optimizing Business Process Execution

- Long running business processes can be optimized with respective to transactions and their boundaries

- Activities can be grouped with other activities within a single transaction
  - Grouping activities in single transaction should be considered carefully

- Activity transaction behavior is set on a per activity basis from within WebSphere Studio on Server tab of activity details
  - Commit Before – Activity will start new transaction
  - Commit After – Activity joins current transaction, commits when completed
  - Participates – Activity joins current transaction or creates new transaction
  - Requires Own – Activity starts and completes transaction
Optimizing Transaction Boundaries

- All activities set to Requires Own transactional behavior
Optimizing Transaction Boundaries

- **Invoke set to Requires Own**
- **Java snippet (participates) creates a new transaction**
- **Invoke set to Commit After**

Diagram showing transaction boundaries and actions:
- **T0**: Receive
- **T1**: Invoke
- **T2**: Script
- **T3**: Invoke
- **T4**: Invoke

<table>
<thead>
<tr>
<th>Java snippet (participates)</th>
<th>Creates a new transaction</th>
<th>Invoke set to Commit After</th>
</tr>
</thead>
</table>

**Transaction Boundaries**
- **T2**:爆破

**Diagram Details**
- Business Reason: Yes
- Transaction: Commit (before, after, durable)
- Requires Own Transaction

**Diagram Elements**
- **Receive**: T0
- **Invoke**: T1
- **Script**: T2
- **Invoke**: T3
- **Invoke**: T4

**Diagram Annotations**
- CC: Commit
- CC: Complete

**Diagram Notes**
- Java snippet participates in a new transaction scenario.
- Commit After indicates the transaction boundaries.

**Diagram Legend**
- NewSnippet
Sequence and Flows

- Sequences execute activities in a single thread model

- Flows allow for parallel processing
  - Conditions can be set on links to specify the logic for the flow
  - If the business process is NOT long-running, the execution will be single threaded
Reversal of a business process - Compensation

- What do you do when you need to undo your (interruptible) process?
  - Order cannot be fulfilled (e.g., production problems)
  - Service called didn’t respond for a too long time
  - Loan approval takes several days and late arrival of new customer credit record data

- All of these share the characteristic of spanning multiple operations over time and the operations that need to be undone are outside the scope of the traditional transaction boundary. (e.g., they were already committed)

- Reversal activities are used to trigger ‘undo’ activities for the Business Process

- They perform a sort of opposite activity
  - Items in order returned to shelf
  - Reservations are canceled

- Supported for multiple activities with the Business Process

• With long running business processes, each activity in the process is a complete step and can be considered a committed act without the possibility of undoing. Consider a product ordering business process. The order is received from the customer and is filled. While the order is being filled, the customer is billed. Before the order is shipped to the customer, however, a cancellation notice is received from the customer, which is considered a business process level failure. At this point, there are a number of completed activities which can not be simply undone with the flip of a switch. In many cases another business process or set of activities must be completed to undo the activity. Compensation is the term for undoing completed activities in a long running business process. With Compensation, you can specify a service to be called should a business process end in a failure. The service will perform some type of undoing to compensate the completed activity.

• The undoing service must be defined by the business process developer and in many cases will be another business process. It may have to perform many different activities to compensate the completed activity. In the case of the order process, suppose that when the customer order was being filled, the shelf stock was lowered below an acceptable level. This caused an order to be placed with a supplier for more products to be shipped. When the customer’s order is cancelled, not only will you need to place the items back on the shelf but also cancel the order with the supplier.

• Note: this is implemented as a BPEL extended feature; there will be a session on Fault handling and Compensation later in the educational program.
Auditing Business Processes

If a Business Process or activity is marked as Business Relevant, then events will be generated by the State Observer. The types of events that can be generated depend on the Construct marked Business Relevant. Here is a list of the events that can be generated for processes and activities and whether or not they are activated by default. These events are stored in the BPEDB.
Summary

- Business Process Choreography Overview
- Business Process Execution Language (BPEL) specification
- Product and tooling support for Process Choreography
- Architecture of the Business Process Container
- Basic Business Process Concepts
- Controlling Business Process Execution