WebSphere® Business Integrator

Process Broker Services Concepts Guide

Version 2.1

WebSphere® Business Integrator

Process Broker Services Concepts Guide

Version 2.1

Note

Before using this information and the products it supports, read the information in “Notices” on page 49

Second Edition (December 2001)

This edition applies to Version 2.1 of IBM WebSphere Business Integrator (product number 5724-A78) and to all subsequent release and modifications until otherwise indicated in new editions.

Contents

Figures

Tables

Preface

This guide explains the concepts behind Process Broker Services for IBM® WebSphere® Business Integrator. Additionally, this guide provides information on a sample that uses Process Broker Services.

Who Should Read This Book

This document is intended for the solution developers who are planning to use Business Integrator Process Broker Services as a part of their solution. Before you read this book, it is recommended that you have a working knowledge of Unified Modeling Language (UML) state machines.

How to Send Your Comments

Your feedback is important to help us provide the most accurate and high-quality information. If you have any comments about this book or any other Business Integrator documentation, please fill out the form at the back of this book and return it by mail or fax, or give it directly to an IBM representative.

Documentation Conventions

Throughout this book, the following conventions distinguish different elements of text:

bold	Menu bar and menu choices, push-button names, icons, check boxes, radio buttons, entry fields, and command names
plain text	Window, screen, and dialog box titles, file names, directories, and keyboard key names
italic	First occurrence of words with special meaning, variables, and emphasis
`monospaced`	Output to the display and user input at the command prompt or in an entry field

The mouse buttons are referred to as ″1,″ ″2,″ and ″3.″ (In some applications, they are referred to as ″left,″ ″middle,″ and ″right.″) Whether your mouse is set with the right-handed or left-handed option, mouse button 1 refers to the button under your index finger.

Chapter 1. Introduction

Process Broker Services is a component of the Business Flow Manager in WebSphere Business Integrator. While this guide provides a technical overview of Process Broker Services, details on other components of the Business Flow Manager can be found in the *WebSphere Business Integrator Concepts and Planning* and *WebSphere Business Integrator Run Time* books.

Process Broker Services is an enterprise bean application that runs on a Java 2 Platform, Enterprise Edition (J2EE)-compliant server, such as the WebSphere Application Server. The two principal functions of Process Broker Services are:

- Brokering of multiple business processes; business processes can be encapsulated in various endpoint applications, such as workflow engines and business applications
- Aggregating content from multiple enterprise information systems in the context of the business process and managing shared access to this content based on the roles of participants

These two functions highlight the fundamental business process management capability of WebSphere Business Integrator.

The Business Flow Manager, including Process Broker Services, enables solution template development. Developing with the solution template typically involves the following steps:

- Supplying the business process definitions
- Composing the relevant adaptive documents that support Process Broker Services functions, which involves specifying the valid application states for the aggregated content and the rules for orchestrating the state transitions
- Formulating the necessary business objects that are referenced from the adaptive documents
- Generating the relevant application adapters to communicate with endpoint applications by using messages to represent business data
- Defining the relevant set of messages
- Assembling the integrated user experience through sequencing of adaptive document views; these views dynamically display the aggregated data associated with a particular adaptive document based on the participant's role.

Solution template development is facilitated by WebSphere Studio Business Integrator Extensions (hereafter referred to as Solution Studio). For more information, see the *WebSphere Studio Business Integrator Extensions Developer's Guide* and the Solution Studio online help system.

Chapter 2. Business Flow Manager high-level design

The high-level component architecture of the Business Flow Manager is shown in Figure 1 on page 4. The components include the following:

- WebSphere Workflow Services—A Joint Flow-based API to access Workflow Management Coalition (WFMC) compliant workflow engines, such as MQSeries Workflow.

  The WebSphere Workflow Services component is primarily used as an interface to a workflow engine, such as MQSeries Workflow. It logs a user to the workflow engine and provides the necessary interfaces for Process Broker Services to launch a process, claim an activity, update the activity status upon completion, and query the process and activity details. Logging a user requires that this component interact with the Trust and Access Manager, another Business Integrator component, to obtain Global Sign-On credentials for the user.
- WebSphere Messaging Services—The Java Message Service (JMS) Listener for asynchronous communication by using message queuing that is used for target adapters in the Business Flow Manager.

  The WebSphere Messaging Services component enables applications and other Business Flow Manager clients to communicate with the Business Flow Manager by using messaging (for example, MQSeries). The JMS Listener, upon receipt of a message, spawns the message-driven bean in WebSphere that in turn invokes Process Broker Services, where the business data in the message is used for process brokering.
- Solution Management Services—Audit, exception handling, and monitoring of business processes.

  The Solution Management Services component uses Business Integrator's Solution Manager client for audit and exception logging. The logs are marked with a unique transaction identifier such as the adaptive document ID, which is then used for correlating the related logs. The Solution Manager console, rendered by Business Integrator's Interaction Manager, is used to view the logs generated by the Business Flow Manager.
- Process Broker Services—Process brokering and content aggregation services that use adaptive documents and controllers for state management.

WebSphere Workflow Services, WebSphere Messaging Services, and Solution Management Services form the base of the Business Flow Manager. For more information on these components, see the *WebSphere Business Integrator Run Time* book. Process Broker Services is a Business Flow Manager component that is realized on top of the rest of the Business Flow Manager components;

that is, it uses the services provided by the other three Business Flow Manager components.

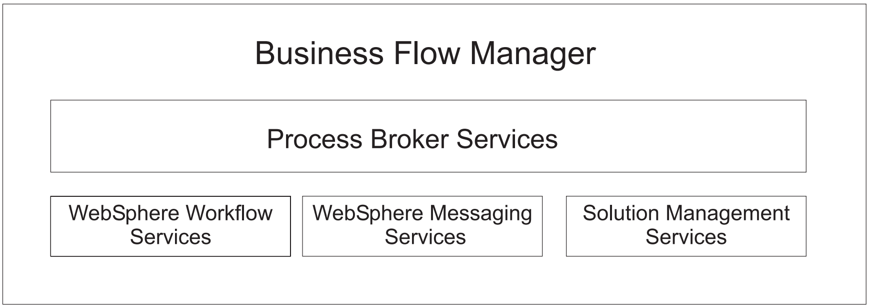
Business Flow Manager
Process Broker Services
WebSphere Workflow Services
WebSphere Messaging Services
Solution Management Services

Figure 1. Business Flow Manager components

Additionally, MQSeries Adapter Builder and MQSeries Adapter Kernel that together make up MQSeries Adapter Offering support Process Broker Services. MQSeries Adapter Builder serves as part of the build-time tool for WebSphere Business Integrator. MQSeries Adapter Builder is used to model and generate microflows; that is, in the context of Process Broker Services, a *microflow* is a composite-command, a sequencing of an ensemble of individual commands that can also, as a collection, act as a single command. A command can be as simple as a method call on a business object.

After the microflow has been generated into source code in MQSeries Adapter Builder and then compiled and run with MQSeries Adapter Kernel, it is called an application adapter, simply referred to as an *adapter*. The adapter forms part of the Information Delivery Manager, one of the managers within WebSphere Business Integrator that supports messaging among the Business Flow Manager, the endpoints and the gateways. The adapter can be viewed as a mechanism that is used by the adaptive documents in Process Broker Services to aggregate content from multiple endpoints or data sources. See "Messaging" on page 21 for more details on how adapters are used in messaging.

After the microflow has been generated in MQSeries Adapter Builder and compiled into an enterprise bean and run in a Java 2 Platform, Enterprise Edition (J2EE)-compliant server, such as the WebSphere Application Server, it is called a Java Service Adapter, as defined in the terminology used in MQSeries Adapter Offering. The Java Service Adapter acts as the composite-command, also called the microflow, under Process Broker Services. See "Commands" on page 13 for more information about commands. See "Chapter 5. Sample application" on page 27 for details on how commands are used in a sample application.

Although it is most common to compile the microflow into an enterprise bean, note that it is also possible to use MQSeries Adapter Builder to compile the microflow into a Java class, such as a MQSeries Adapter Kernel Command Interface adapter.

For more information on MQSeries Adapter Builder and MQSeries Adapter Kernel, refer to the MQSeries documentation listed in “Other Libraries” on page 56.

Chapter 3. Process Broker Services architecture

This chapter explains Process Broker Services architecture, including the communication model, the Process Broker Services interface, and adaptive documents.

Business Flow Manager communication model

Clients to the Business Flow Manager can use either remote method invocation/Internet InterORB Protocol (RMI/IIOP) or Java Message Service/MQSeries (JMS/MQSeries) to communicate with the Business Flow Manager, including the Process Broker Services component. For example, the Interaction Manager is a Process Broker Services client that uses RMI/IIOP to communicate with Process Broker Services to render the dynamic executable content from adaptive documents to users who are interacting via Web browsers. Alternately, an endpoint application can use MQSeries to send messages to Process Broker Services, such as the Open Application Group's Business Object Documents (OAG-BOD), an Extensible Markup Language (XML) application-to-application messaging standard. Gateways, such as Partner Agreement Manager and DataInterchange, can also use MQSeries to send messages to Process Broker Services.

Process Broker Services also uses both RMI/IIOP and MQSeries to communicate to the various enterprise information systems, or endpoints. For example, the Process Broker Services uses RMI to communicate to business objects, such as a Purchase Order or a Ship Schedule. The Process Broker Services can also use the WebSphere Messaging Service and communicate to the endpoints via MQSeries by using OAG-BOD messages. These messages are sent either point-to-point by using MQSeries to endpoints, or via the Information Delivery Manager, a Business Integrator message broker that transforms and routes the messages as warranted. In both cases, point-to-point and using the Information Delivery Manager, Business Flow Manager source adapters must be built using MQSeries Adapter Builder and operated using Information Delivery Manager. These source adapters semantically adapt the business data, aggregated by Process Broker Services from various sources, to messages that are then communicated to the endpoints. A generic target adapter is provided to receive reply messages from endpoints to the Business Flow Manager.

Process Broker Services interface

Process Broker Services provide dynamic business services that are accessible to clients through the Process Broker Services interface. For more information on the Process Broker Services interface, see the *WebSphere Business Integrator Process Broker Services Developer's Guide*. A *business service*, in the context of Process Broker Services, is a set of command operations that is exposed within a business process.

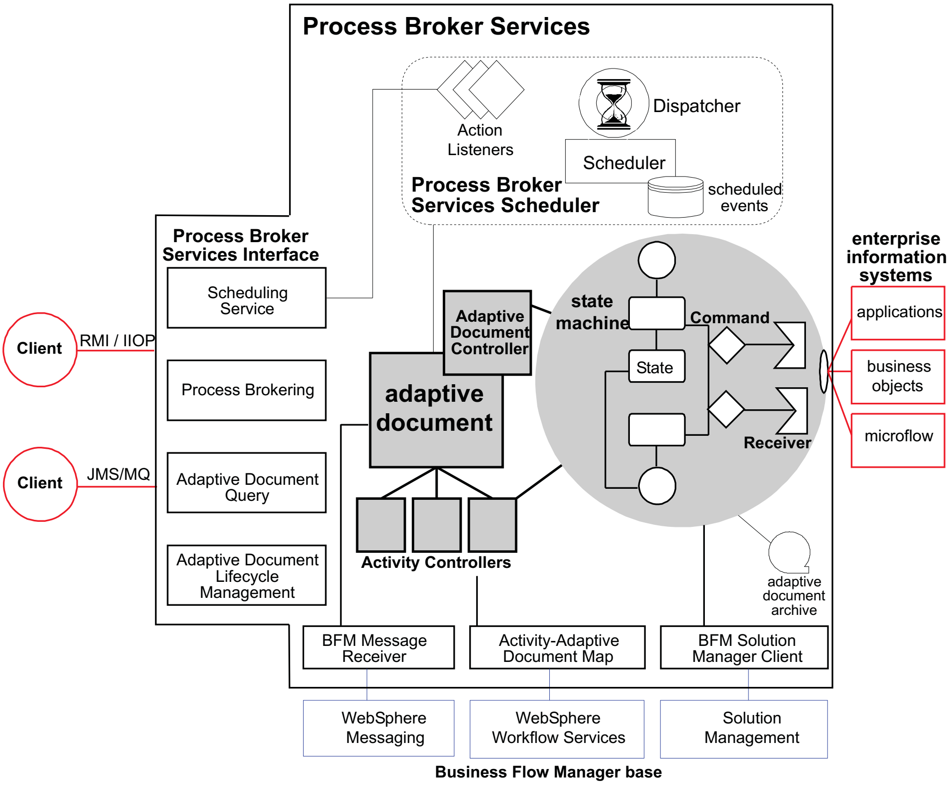

Figure 2. Process Broker Services architecture

Though business services vary depending on the business needs being addressed, all business services can fit into one of the following categories: Process Brokering, Adaptive Document Query, Adaptive Document Lifecycle Management, and Scheduling Service, as shown in Figure 2.

- Process Brokering Service — enables Process Broker Services clients to invoke dynamic business services that are made available based on the business state of the adaptive document. The business services are dynamic because they are state dependent; that is, the available set of services vary with any change in the business state of the adaptive document instances. The client can trigger any service by raising an event against a specific adaptive document instance. The actual invocation of the service is made by a service request on the Process Broker Services with the adaptive document ID, the business event name, and other parameters.

- Adaptive Document Query Service — enables Process Broker Services clients to query the business state of the adaptive document, ascertain the available business services for a given business state, access the business content aggregated by the adaptive document, and query for navigational purposes (for instance, a list of adaptive documents that satisfy a given criteria).
- Adaptive Document Lifecycle Management Service — enables Process Broker Services clients to create, delete, archive, and restore adaptive documents.
- Scheduling Service — enables Process Broker Services clients to automate the service invocation by scheduling it with the Process Broker Services Scheduler. At the scheduled time, the Process Broker Services Scheduler notifies the registered action listener, which in turn makes the service invocation on the Process Broker Services. The Process Broker Services provides one default action listener and a provision for user-defined action listeners. For more information on the default action listener, as well as how to define custom action listeners, see the *WebSphere Business Integrator Process Broker Services Developer's Guide*.

For more information on the invocation methods of each of these types of business services, see the *WebSphere Business Integrator Process Broker Services Developer's Guide*.

Adaptive documents

The *adaptive document* is a component, comprised of an entity bean with container-managed persistence, that links the content aggregated from various data sources to business processes and people. The adaptive document enables collaborative business process management through the orchestration of a variety of applications and user interactions in the context of a business process. Figure 3 on page 10 provides a conceptual view of an adaptive document.

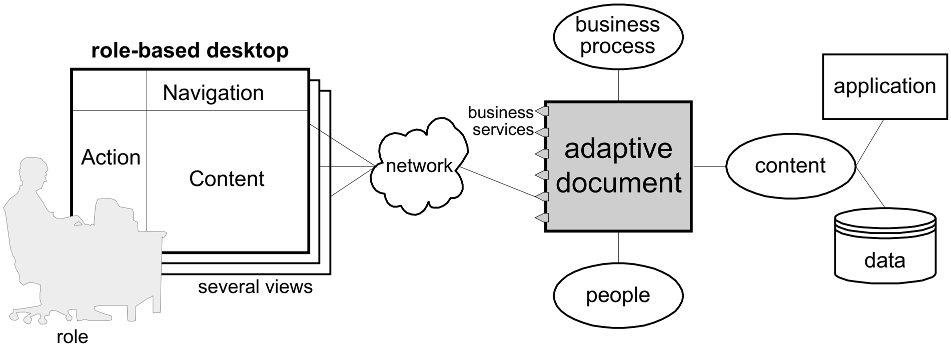

Figure 3. Conceptual view of an adaptive document

An example of an adaptive document is the collaborative forecasting and planning process. The collaborative business process management experience generated by the adaptive document is characterized by four factors:

- The ability to share information (for example, historical sales that are collected from an ERP system)
- The ability to share appropriate decision support tools (for example, planning engines, such as a forecasting application whose services are brokered) to act on shared information
- The business events that define the context in which the information and the tools are shared (for example, a large customer order event that triggers the collaborative planning process)
- Enabling collaboration among appropriate role players (for example, demand planner, supply planner, parts supplier) in the business context by using the information and the tools that are aggregated and shared by the adaptive document

The adaptive document facilitates the collaborative experience by dynamically exposing a set of business services based on who you are (role) and where you are in the process (process context). Invoking a business service exposed by an adaptive document is essentially executing a business transaction. The ensemble of business transactions, in turn, constitutes the business conversation; an adaptive document can support multiple business conversations that are part of any collaborative business process management.

A business transaction can have multiple levels (for instance, a long-running collaborative forecasting process, collaborative planning activity of building a forecast, atomic transaction of getting the sales history of a specific item) and span multiple parties (for instance, two decision makers or applications from different organization units within an enterprise or two decision makers or applications from different enterprises). The adaptive document is capable of executing these multilevel, multiparty business transactions. This capability enables an adaptive document to do business process brokering and deliver the right information and right tools to the right group of people in the context of executing a collaborative business process.

The adaptive document can also be understood from the interaction experience of a user in the collaborative solution. Users in this case interact through views of an adaptive document. Any particular view has three components (collectively called the role-based desktop): Action, Navigation, and Content (see Figure 3 on page 10).

- The Action component corresponds to the dynamic business services that are exposed by an adaptive document and is invoked via the Process Broker Services interface. The Action component changes based on the role of the user and the context of the business process.
- The Navigation component can be either process-centric (for example, displaying the set of actions given the current process state) or document-centric (for example, displaying the list of adaptive documents that require some action). The adaptive document supports both navigation models. The Navigation component is accessed via the Process Broker Services interface.
- The Content component is marshaled by the execution of the business services in an adaptive document; that is, the business transactions are brokered by the adaptive document controllers to aggregate the content from a multitude of back-end sources (such as applications, business objects, and databases).

The different views and the role-based desktop are handled by the Interaction Manager; see the *WebSphere Business Integrator Run Time* book for details.

Activity and adaptive document controllers

The adaptive document uses the *controller*, a state machine, to implement the brokering functions. A controller is essentially a services broker. When a service request comes to the adaptive document, it uses the application state to determine whether the request can be entertained, and, if so, it uses the controller to broker the services needed to satisfy the request. The design characteristics for the service broker, or controller, are:

- The state transitions are transactional. All the commands that are triggered by a service request need to be executed as a logical unit of work. If there is a failure, the state change does not occur and, if necessary, recovery procedures, such as a compensation script, are executed.
- It is possible to execute an ensemble of commands as part of a state transition. Such an ensemble can be either a simple sequence or multiple commands joined together as a microflow.

- The controller defines the dynamic behavior of the adaptive document. It is possible to modify the definition of the controller, publish it, and dynamically change the behavior without disrupting or shutting down Process Broker Services.
- Process Broker Services facilitates dynamic e-business systems, in which the service providers for the individual commands in the controller can be dynamically mapped. This implies a separation between the commands and the receivers that implement the commands.
- The controller can schedule triggers that generate service requests on the controller. This autonomy is essential to capture timeouts and other temporary constraints that define the dynamic behavior of the service broker.

An adaptive document instance has an adaptive document controller and zero or more activity controllers; the number of activity controllers is determined by the number of activities associated with the adaptive document. The activities in business processes are defined in the workflow engine and accessed using WebSphere Workflow Services. The adaptive document controller and the activity controller have an identical structure, as shown in Figure 4.

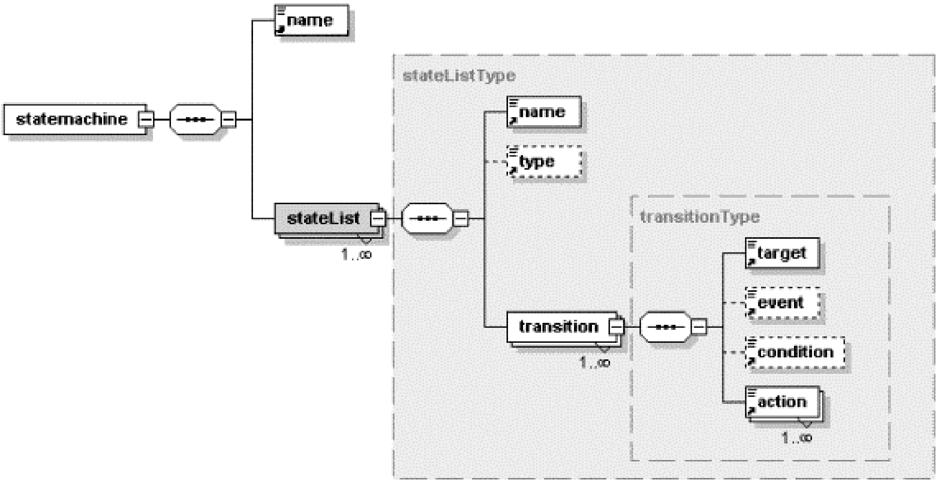

Figure 4. Structure of a controller

The controllers are defined in XML and the Process Broker Services read these definitions at runtime. Changes can be made to the XML definitions at any time and the Process Broker Services refresh the in-memory definition of the controllers based on the new definitions.

The controller is a named state machine. The state machine consists of:

- The list of states that are named and also identified based on their type (such as normal and terminal).
- The permissible transitions between these states where each transition is specified from a given state to a target state. There can be many transitions from any given state.
- The state transitions are triggered for defined events when a given condition expression on the transition is satisfied.
- When a state transition occurs, one or more actions or commands are executed. These actions or commands can be sequenced in any fashion and such a sequence of actions is referred to also as a microflow.

The business state is a combination of the state of the adaptive document, managed by the adaptive document controller, and the state of any other workflow activity in which the adaptive document is a participant, managed by the appropriate activity controller.

Commands

Commands can be viewed as interfaces to the business logic, or interface definitions to the various endpoints that are engaged in business process management. Using the composite design pattern, shown in Figure 5, it is possible to compose commands to form composite-commands, also called microflows. MQSeries Adapter Builder can be used to sequence commands and construct microflows.

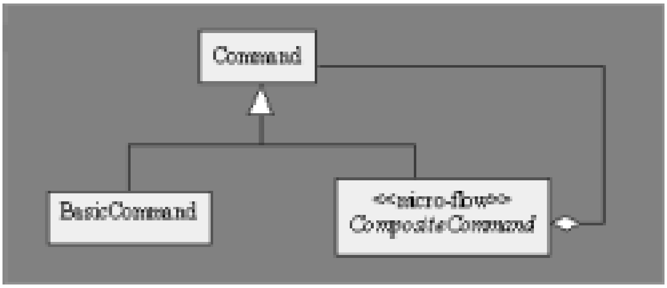

Figure 5. Composite design pattern for commands

The commands are specified in the controller by using XML. The XML command data structure is shown in Figure 6 on page 14.

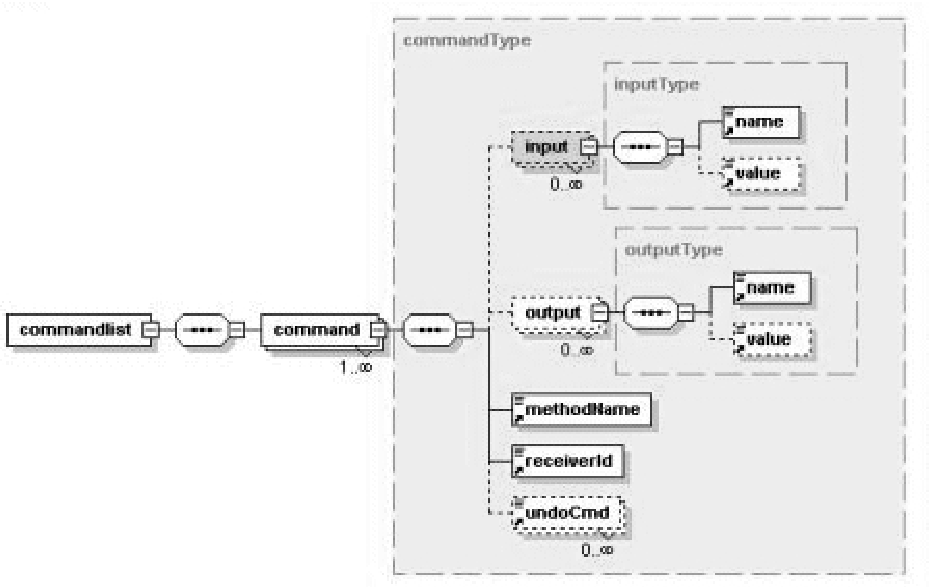

Figure 6. Structure of commands

Each command has an input and an output data structure where the individual attributes can be specified as name-value pairs. The command is identified by its method name. The commands executed within a state transition are in a single transaction. In the event of a transaction failure, the actions executed by the command are undone using the **undo** command. Some of the endpoints that are engaged can not be transactional systems and recovery entails using compensation logic. Such logic can be encapsulated in the **undo** command. The receiver associated with a command is identified by the receiver ID. For more information on commands, see the *WebSphere Business Integrator Process Broker Services Developer's Guide*.

Receivers

A *receiver* is an interface to the service providers, or implementations of the business logic expressed in the commands. Receivers enable the dynamic mapping of the service providers to the commands. There can be many types of receivers, such as a JMS receiver for asynchronous connectivity to various back-end applications and systems, or a RMI receiver for synchronous connectivity to other business objects and applications.

MQSeries Adapter Builder, used to build the microflow, can be used to build specialized receivers that implement the microflow. These receivers are realized either as Java objects or session beans.

All receivers in Process Broker Services are specified by using XML. The structure of the receiver is shown in Figure 7.

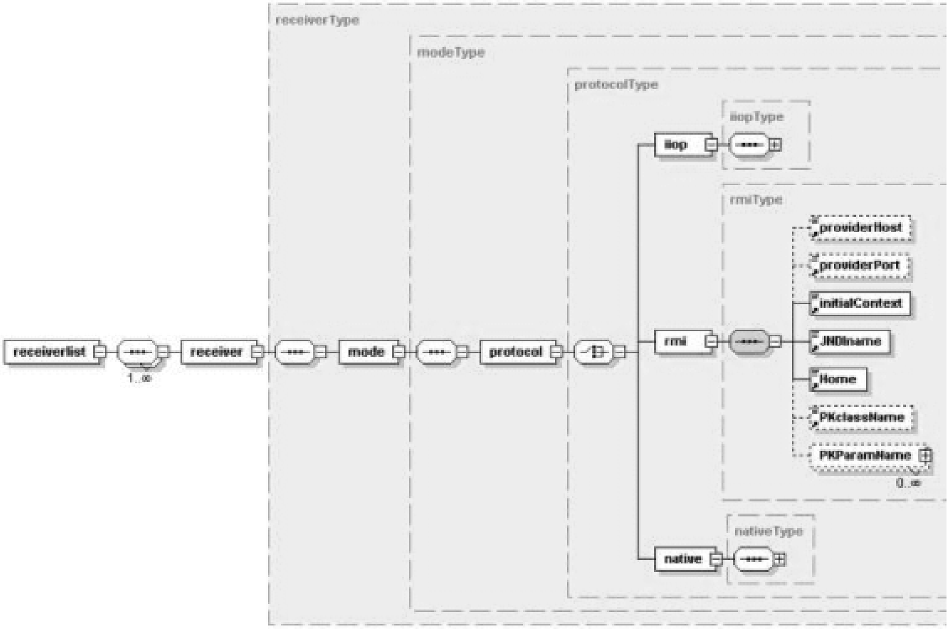

Figure 7. Structure of receivers

Identifying the protocol type and the relevant parameters for establishing the connectivity and invoking the method name specifies the receiver. The RMI protocol is used to identify the Java class that is instantiated in a different Java Virtual Machine (JVM) than the Process Broker Services. The native protocol is used when the Java class that is engaged is in the same JVM as the Process Broker Services. Receivers can be added with other protocols such as JMS and IIOP. For more information on receivers, see the *WebSphere Business Integrator Process Broker Services Developer's Guide*.

Process Broker Services scheduler

The Process Broker Services scheduler enables time phased automatic invocation of service requests. This provides the ability for the adaptive document and its associated controllers to trigger events automatically to drive state transitions. Such a capability is very useful in modeling timeout events, for example; in this case, the timeout event is scheduled by a command in a controller and the event is triggered on schedule on the controller.

The Process Broker Services scheduler collaborations are shown in Figure 8 .

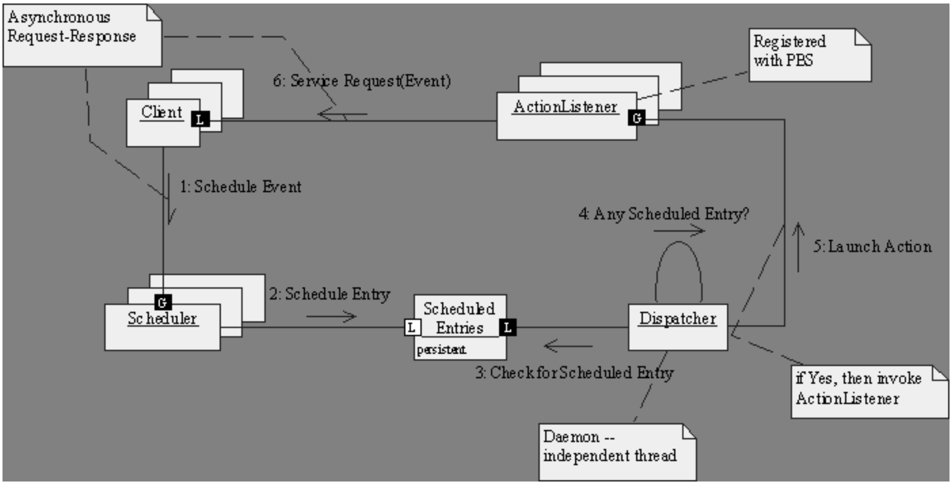

Figure 8. Request-response collaboration for a scheduled event

1. Clients, typically an adaptive document and associated controllers, schedule an event by making a request on the scheduler. The scheduler is globally visible to the client as a well-known entity. This request is asynchronous in that the event is scheduled and the client is not blocked.
2. The scheduler commits the scheduled event, where the scheduled entries are persistent.
3. The dispatcher, running as a separate thread of execution or daemon, periodically checks for entries to act upon.
4. The time between dispatcher checks for entries can be customized.
5. The dispatcher launches the action listener registered for a particular scheduled event. Process Broker Services comes with a number of standard action listeners. There is also provision in the Process Broker Services scheduler to have a user-defined action listener. An action listener is a handler that, upon notification, performs service-related tasks.
6. The action listener makes the appropriate service request to complete the asynchronous request-response cycle for a scheduled event.

Adaptive document archive

The adaptive document archive provides a transitive closure mechanism for adaptive documents. When an archive request is made, the adaptive document, associated controllers, and object references are all serialized and persisted. The adaptive document can then be revived through a restore request. The archive and restore requests can be made as commands from the controllers.

This feature is very useful as an adaptive document represents a business transaction. Business processes can require archiving business transactions upon completion for non-repudiation purposes. This is also useful in the case of modeling process brokering for long-running transactions where the time scales can be weeks or months.

Chapter 4. Integration with Business Flow Manager

The base components of Business Flow Manager, shown in Figure 1 on page 4, include the WebSphere Workflow Service, the WebSphere Messaging Service, and the Solution Management Service. Process Broker Services uses all three of these services.

Workflow

The WebSphere Workflow Service provides a Joint Flow-based API to access the workflow engine, such as MQSeries Workflow. Process Broker Services invokes workflow commands, such as method invocations on WebSphere Workflow Service, to launch a business process, claim an activity in a process, indicate completion of an activity, and inquire about the status of either a process or an activity. Process Broker Services associates adaptive document instances with activities. This association is captured in the activity-adaptive document map (AA map). An adaptive document instance can participate in multiple activities either within the same process instance or multiple process instances.

A new activity becomes available, as specified in the process definition, when the business process is updated in the workflow engine to indicate the completion of a prior activity. The availability of a new activity results in the update of the AA map. Process Broker Services then launches an activity controller for the adaptive document instance associated with the activity. The activity controller brokers the collaborations, or business conversation, necessary to execute the activity. The activity controller issues workflow commands on WebSphere Workflow Service as part of the controller definition to claim the activity and to indicate completion of the activity. A number of application-specific states are typically involved between the claim and complete events. The activity controller can engage humans as well as applications in the execution of the activity.

The interaction between Process Broker Services and the Workflow Engine is shown in Figure 9 on page 20.

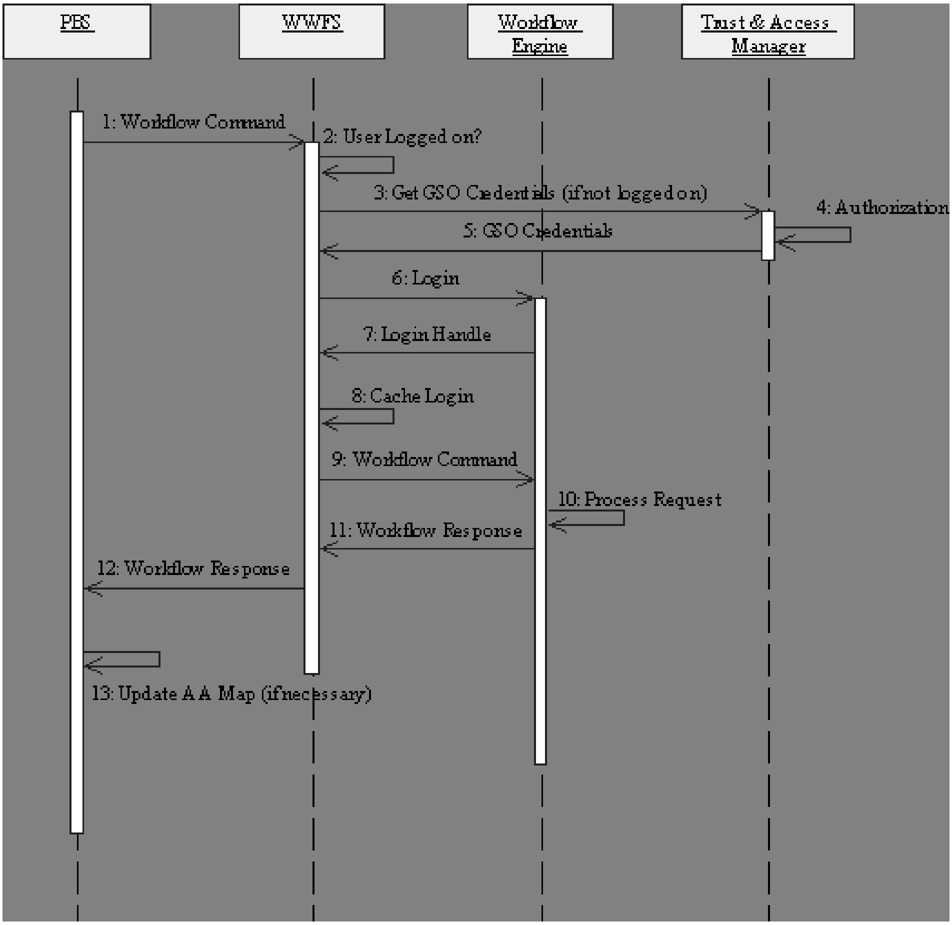

Figure 9. Process Broker Services and workflow interaction

1. Process Broker Services creates a workflow command. Typically, the controllers create this command.
2. WebSphere Workflow Service checks to see if the user has already logged on to the workflow engine. This is necessary especially for launching processes, claiming, and completing an activity because only users with certain privileges are authorized to perform such tasks.
3. If the user is not logged on to the workflow engine, the WebSphere Workflow Service makes a request to the Trust and Access Manager, a Business Integrator component, to obtain the Global Sign-On credentials. A Global Sign-On is a mechanism to perform credential mapping for a user with multiple identities associated with various endpoint applications.

4. The Trust and Access Manager makes an authorization check to see whether the user does have access to the requested system.
5. The Trust and Access Manager returns the authorization credentials to WebSphere Workflow Service.
6. The WebSphere Workflow Service uses the Global Sign-On credentials to logon to the workflow engine.
7. The workflow engine upon successful logon returns the logon connection handle.
8. The WebSphere Workflow Service caches this logon handle for purposes of optimizing the connection to the workflow engine.
9. The WebSphere Workflow Service then forwards the workflow command requested in step 1.
10. The workflow engine processes the request.
11. The workflow engine response is received by WebSphere Workflow Service.
12. The WebSphere Workflow Service forwards the workflow response back to Process Broker Services.
13. Process Broker Services updates the AA map if necessary (especially when the state of an activity changes).

Messaging

Process Broker Services utilizes the WebSphere Messaging Service (see Figure 1 on page 4) to receive and send messages. The messages are typically in Extensible Markup Language (XML) format, like the Open Application Group's Business Object Documents (OAG-BOD) messages. The recommended transmission protocol is Java Message Service (JMS), though the MQSeries Point-to-Point (MQPP) protocol can also be used.

Messaging is performed by the Information Delivery Manager, one of the managers within WebSphere Business Integrator. The Information Delivery Manager can perform assured delivery of messages, transformation of data elements and related functionalities, routing of messages, and message brokering. The components of Information Delivery Manager that actually perform the functionality described include: MQSeries Adapter Kernel using adapters built with MQSeries Adapter Builder, MQSeries for assured delivery, and optionally MQSeries Integrator for message brokering services such as complex routing, data transformation, and data mediation.

The following collaboration illustrates how business data is communicated by using messaging from any endpoint to Process Broker Services. Though this is not shown in the following collaboration, Process Broker Services can also send business data as messages to any endpoint.

The collaborations associated with an incoming message are shown in Figure 10.

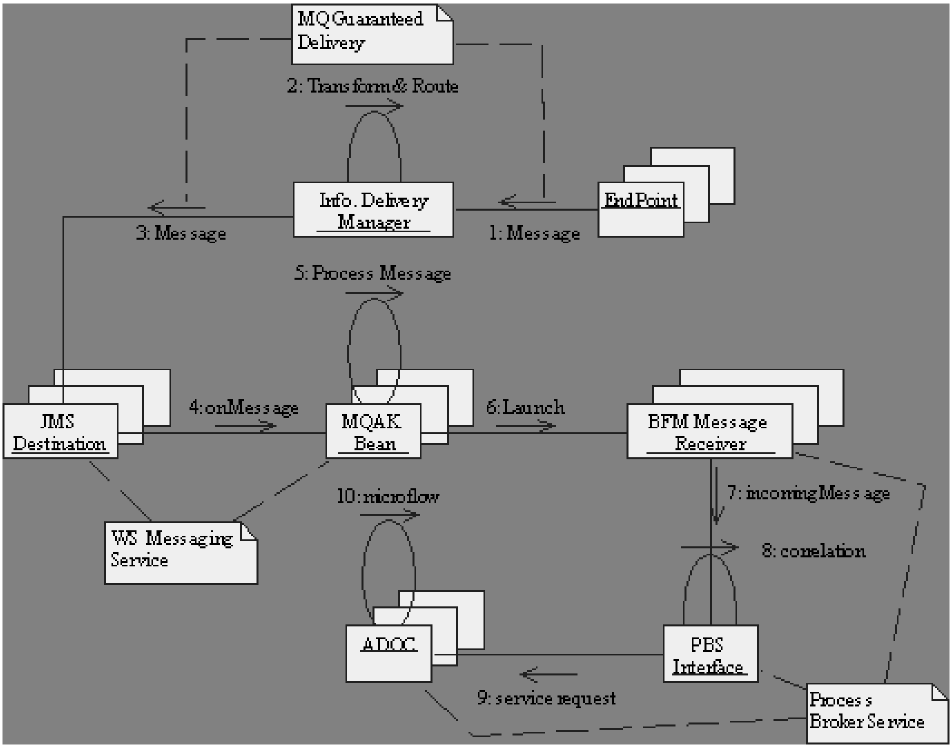

Figure 10. Process Broker Services incoming message

1. An endpoint, such as an ERP system or a scheduling application, sends a message. These messages are sent by using MQSeries and have guaranteed delivery. Typically, an MQSeries adapter at the endpoint semantically adapts the message from the application output to the standardized form, such as OAG-BOD. The message is also enveloped with JMS and Business Integrator headers.
2. The Information Delivery Manager, acting as the message broker, receives the message, transforms it as necessary, and routes the message. The source and the target for the messages are essentially decoupled by the Information Delivery Manager. This mechanism provides loose coupling and extensibility of the application integration.
3. Because the message is intended for Process Broker Services, it is routed to a JMS destination. The JMS destination can be either a specific queue (as in point-to-point messaging) or a topic (as in publish-subscribe messaging).

4. The MQSeries Adapter Kernel bean that implements the JMS Listener's onMessage method receives the message.
5. The MQSeries Adapter Kernel bean processes the message; that is, it unwraps the JMS message header and formats it as necessary.
6. The MQSeries Adapter Kernel bean then launches the Business Flow Manager message receiver. The Business Flow Manager message receiver is similar to a thin MQSeries application adapter that is designed to handle all incoming messages (of any type) to Process Broker Services.
7. The Business Flow Manager message receiver makes an incoming message call to the Process Broker Services interface. This call also passes the message to the Process Broker Services interface.
8. The Process Broker Services interface does the necessary correlation. If an adaptive document instance is referred to in the message header, then that is used for correlation; otherwise, a new adaptive document instance is created and the message is associated with that instance.
9. The Process Broker Services interface makes a service request on the correlated adaptive document. Process Broker Services brokers the service-request appropriately to either the adaptive document controller or one of many activity controllers.
10. The service request triggers the appropriate controller state transition, resulting in the execution of a transaction consisting of one or more commands or microflows within the controller.

Solution management

The Business Flow Manager Solution Manager client in Process Broker Services uses the Solution Management Services, a Business Flow Manager component, to generate audit logs and exception logs that are persisted in the Solution Manager. The audit and exception log messages are XML messages that are sent to the Solution Manager by using MQSeries. The Business Flow Manager Solution Manager client can be invoked as a command within the adaptive document controller or activity controllers in Process Broker Services to initiate audit and exception logging. Process Broker Services automatically logs the adaptive document transaction history.

The adaptive document is a business entity that enables the execution of process driven business transactions. An e-payment adaptive document, for example, transcends the various steps in an e-payment business process. Such an e-payment business process in turn engages multiple applications. From a solution management perspective, it is important to be able to view the overall transaction history of the adaptive document.

The adaptive document controller manages the state transitions. The controller, upon receiving a valid event in a given state, triggers a permissible state transition. The transition in turn launches one or more actions. These actions are subtransactions against various endpoints. The adaptive document transaction history consists of a record of the states that the adaptive document has traversed and the time in and time out for each individual state in the traversed state list. The transaction history for an adaptive document is uniquely associated with the adaptive document ID that represents the transaction identifier.

The log message fields and mapping to the adaptive document status, provided automatically by the Business Flow Manager Solution Manager client, is shown in Table 1:

Table 1. Solution Manager log messages

Message Field	**Values**	**Solution Manager Log Field**	**Description**
SourceID	Business Flow Manager App ID (for instance, Business Flow Manager or Business Flow Manager1)	MSG_SRC_ID	The Business Integrator application name
Body Category	adaptive document	MSG_BDYCAT	The category of the log event
Body Type	″Entry, Exit″	MSG_BDYTYPE	The type of event being logged for this category
Transaction ID	adaptive document ID	MSG_XACTION_ID	The key for the entry
Body Data	See Table 2 on page 25	SRC_MSG	The remaining information for this category of event; it has a DTD unique to this category. The body data is stored in this column.

The message fields and values for the body data are shown in Table 2 on page 25:

Table 2. Body data messages

Body Field	**Values**	**Description**
ACTIVITYID	Activity ID	Activity ID, if applicable
ADOCTYPE	Adaptive document type	The type of adaptive document being logged such as ″e-Payment″
USERID	User ID	User ID, if applicable
EVENTNAME	″Trigger event name″	Name of the event that caused the transition
DATETIME	Log event time	The log event time (Java date)
FROMSTATE	Adaptive document state name	The name of the prior state
TOSTATE	Adaptive document state names	The name of the next state

Chapter 5. Sample application

This chapter describes a sample application that demonstrates the use of Process Broker Services in WebSphere Business Integrator. The chapter provides a description of the application, introduces a methodology to design the Process Broker Services artifacts needed to implement the solution, and illustrates how to derive the artifacts by using this methodology.

In the context of this sample, solution design generally follows this method:

1. The information model, the organization model, and the business process model are defined. For more information on these models, see "RFQ application" on page 28.
2. Using the information model and the business process model, the number of adaptive documents needed in the system is identified. To learn more about adaptive documents, see "Adaptive documents" on page 9.
3. Using the business events and their prerequisites, the adaptive document state machines are designed. To learn more about adaptive documents, see "Adaptive documents" on page 9.
4. Using the processing rules associated with these business events, the commands are identified that need to be executed as part of state transitions.More about commands is explained in "Commands" on page 13.
5. Macro workflows are used to define the processing rules that dictate collaboration with user or software actors in the system; macro workflows outline the activities involved in a business process and are implemented by adaptive document controllers. A state machine models an adaptive document controller's behavior and commands affect the behavior. To read more about workflows, see "Macro and micro workflows" on page 36; to read more about controllers, see "Activity and adaptive document controllers" on page 11.
6. Micro workflows are used to define the microflows that complete the activities outlined in the macro workflows; micro workflows are implemented by activity controllers, which are designed and defined the same as adaptive document controllers. To read more about workflows, see "Macro and micro workflows" on page 36; to read more about controllers, see "Activity and adaptive document controllers" on page 11.

It is important to note that although an MQSeries Adapter Offering source adapter can serve as a receiver for message-based integration with endpoint applications, endpoint integration via MQSeries adapters is not illustrated in this sample.

RFQ application

A Private Trading Exchange (PTX) enables collaboration between the trading partners of a company and its employees in the context of order logistics management. The scenario follows the life cycle of a Request for Quotation (RFQ), starting from creation and ending with the completion of the vendor selection. The application has three parts:

1. Information model — This describes the underlying data structures the system creates, reads, updates, or deletes, and their relationships.

   In this sample, there are two primary data structures: the RFQ and the Quote. More information about these primary data structures can be found in "RFQ Process macro workflow" on page 37 and "Quote Creation Process macro workflow" on page 42. A realistic RFQ application needs many more supporting data structures, but these are not discussed in this sample because they are irrelevant from the process brokering point of view.
2. Organization model — This organizes the users of the system by the business roles they play.

   In this sample, the organization model consists of the PTX organization and various seller organizations. The roles in the PTX organization include Buyer and Buyer-Approver. The roles in a typical seller organization include Seller and Seller-Approver.
3. Business process model — This describes the business events that are received by the system, as well as those that are generated by the system. For incoming events, the model will describe the business rules that are applied for processing the events. For outgoing events, the model describes the business rules that govern the generation of these events. Business events are differentiated from workflow events. All workflow events are described as part of the business rules that generate or consume business events.

   In this sample, Figure 11 on page 29 shows the business events in the RFQ application. Workflow events are not addressed in this sample because those are part of the processing rules associated with the business events. An example of a workflow event is a Buyer-Approver acting on a submitted RFQ to approve or reject it. This event is captured as part of the business rules specified to process a submitted RFQ.

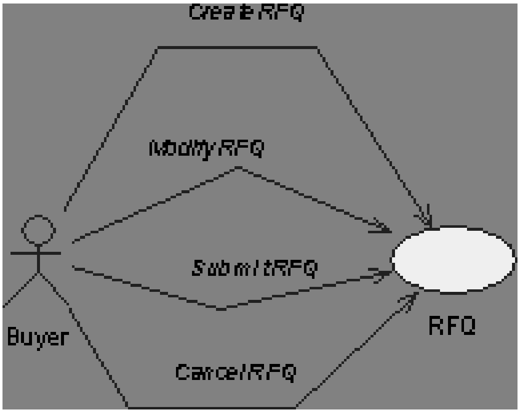

Figure 11. Business events in an RFQ process

Business events

The business process model defines the business events that are part of the system. Business events are events within the context of the business process generated by actors, such as humans or endpoints; examples of business events include a human submitting an RFQ or an endpoint responding to Process Broker Services via messaging.

Note that while there are two ways to send business events to Process Broker Services (via synchronous method invocation over Internet InterORB Protocol (IIOP) or via asynchronous messaging through the Java Message Service (JMS) Listener), the solution design remains the same regardless of the protocol used.

The business events in this sample include Create RFQ, Modify RFQ, Submit RFQ, and Cancel RFQ. These business events and their characteristics are explained in Table 3.

Table 3. Business events and their characteristics

Business Event	Source	Preconditions	Processing Rules
Create RFQ	Buyer	None	Persist RFQ data in database.
Modify RFQ	Buyer	RFQ created but not submitted or cancelled.	Modify RFQ data in database.
Submit RFQ	Buyer	RFQ created but not cancelled or submitted.	Start the RFQ process. See Figure 12 on page 31.
Cancel RFQ	Buyer	RFQ created.	If RFQ Process is active, terminate it. Notify sellers and approvers.

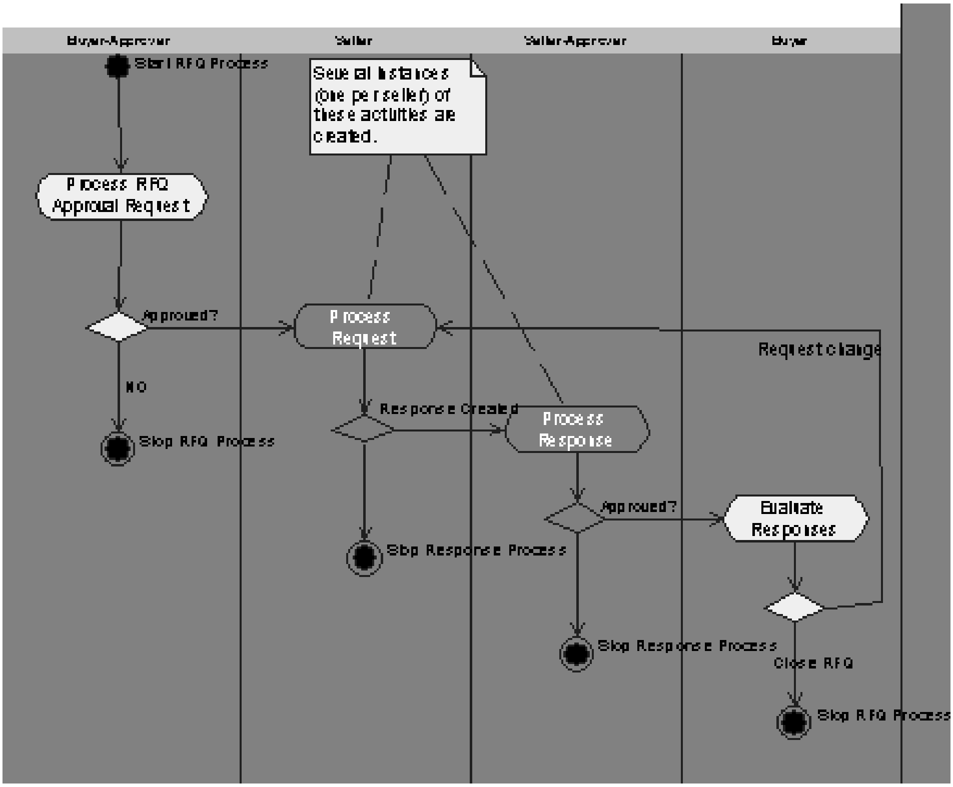

Figure 12. RFQ process

Solution design

The key design artifact of a Process Broker Services-based solution is the adaptive document. Adaptive documents provide the process brokering capability by intercepting business events and servicing them based on the application state; therefore, identifying where adaptive documents are needed in the application is the main step in solution design.

To apply the adaptive document concept effectively, guidelines are used to identify how many and where adaptive documents are needed in a solution. It is important to note that an adaptive document requires a complex design pattern and identifying the need for an adaptive document does not follow a definitive path; the following guidelines serve only as suggestions for locating the need for an adaptive document:

1. Define all the relevant business objects in the business problem; the information model from the analysis phase helps in identifying the business objects.
2. Identify the key business objects that can serve as the handlers of the business events identified in the business process model.
3. Begin identifying the needed adaptive documents by noting the number of business objects highlighted in step 2; typically there is a one-to-one correspondence between these key business objects and the adaptive documents in the solution.

Using the RFQ process in the PTX as an example, all business events specified in the business process model are in the context of the RFQ business object; therefore, only one adaptive document type is needed in this application, an RFQ adaptive document that handles these business events.

After the adaptive documents are identified, the next step is to define them for the solution. By defining an adaptive document, the collaborative behavior that the adaptive document encapsulates is defined. The behavior of the adaptive document is defined by using a state machine combined with a command design pattern (for more information on the command design pattern, see *Design Patterns: Elements of Reusable Object-Oriented Software* published by the Addison-Wesley Professional Computing Series). The state machine is defined as a set of finite states and the transitions permissible at each state. For each state transition, there is an associated event that triggers it. The state transitions are triggered by a service request made by a client on the adaptive document. As part of the service request, the client passes an event identifier, a set of input parameters, and a context to the adaptive document. As part of a state transition, the adaptive document executes a set of commands as a transaction. Because the commands are designed using the command design pattern, the actual command work is delegated to a receiver.

To define the state machine and adaptive document, begin with the business process model to identify the business events that are handled by this adaptive document. These business events map to the events that drive the adaptive document state transitions. The business rules that govern the processing of these events are used to identify the state transitions, the commands that need to be executed for each transition, and the receivers of these commands.

The state machine that defines behavior mainly holds state information; however, an adaptive document also holds minimal state information. This includes the current state of the adaptive document and pointers to the business objects referenced by it. An adaptive document only needs to reference the top-level business objects, as there is no need for the adaptive document to reference business objects navigable from the top-level business objects.

RFQ adaptive document design

This section presents the design of the RFQ adaptive document for the sample application. The RFQ adaptive document entity is used to hold the minimal state information, and the RFQ adaptive document controller is used to define the behavior.

RFQ adaptive document entity

The design of the RFQ adaptive document entity is shown in Figure 13.

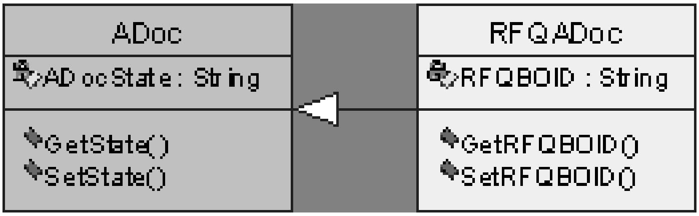

Figure 13. RFQ adaptive document entity

The adaptive document entity extends the generic adaptive document class to hold application-specific data references.

RFQ adaptive document controller

Figure 14 on page 34 shows the state machine of the RFQ adaptive document controller.

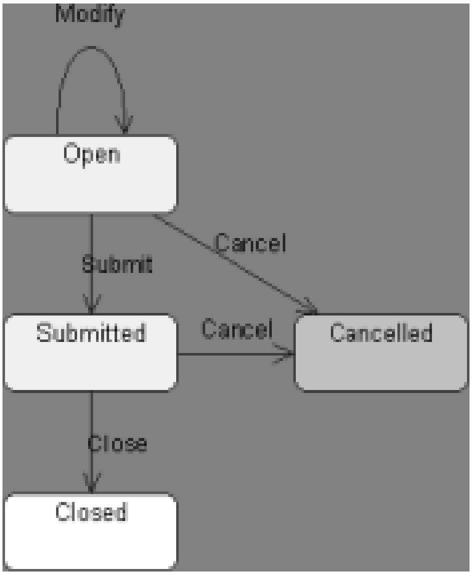

Figure 14. RFQ adaptive document controller state machine

Note that the states cancelled and closed have no transitions defined for them. The cancelled state is denoted as a terminal state. RFQ adaptive documents that are cancelled are periodically purged by the system management services of Process Broker Services. RFQ adaptive documents that are closed are archived periodically by Process Broker Services as well. See the *WebSphere Business Integrator Process Broker Services Developer's Guide* for details on system management services.

Table 4 on page 35 shows the commands that are executed as a part of state transitions.

Table 4. Commands executed as state transitions

From	To	Event	Commands
Open	Submitted	Submit	StartRFQProcess
Open	Cancelled	Cancel	DeleteRFQData
Open	Open	Modify	ModifyRFQData
Submitted	Closed	Close	• CreateNotificationInfo • Notify • CreatePO • DeleteRFQData
Submitted	Cancelled	Cancel	• TerminateAllProcesses • CreateNotificationInfo • Notify • DeleteRFQData

Commands in adaptive document controller

Table 5 on page 36 shows the details of the commands and their corresponding receivers. System commands are indicated by an asterisk (*).

Table 5. Adaptive document controller commands

Command	Receiver	Description
StartRFQProcess*	WWF Services	Instantiates and starts the flow graph shown in Figure 15 on page 38.
DeleteRFQData	RFQ BO	Release persistent storage associated with the RFQ.
ModifyRFQData	RFQ BO	Modify persistent RFQ data.
CreateNotificationInfo	RFQ BO	Generate information on whom to notify and what to notify.
Notify*	Notification BO	Send email notifications based on information collected by the previous command.
CreatePO	Process Broker Services Scheduler	Create PO adaptive documents and start PO business process asynchronously.
TerminateAllProcesses*	BFMAdmin	Terminate RFQ and Quote Creation processes. Terminate all associated processes of an adaptive document instance.

Macro and micro workflows

The processing rules specified for a business event can dictate that certain business events necessitate collaboration among the actors in the solution. The actors can be human role-players or application actors, such as endpoints. Typically, this collaboration entails a long-running process that can span several days or even months. In a Business Integrator solution, this collaboration is captured in a macro workflow graph. A macro workflow graph is a directed graph, where the nodes denote activities that are to be completed by designated actors and arcs denote the collaboration pattern. Each activity can be defined in a micro workflow. In the context of Process Broker Services, a *macro workflow* is implemented by an adaptive document controller and a *micro workflow* is implemented by an activity controller.

Micro workflow refers to the detailed activity flow within a macro activity. A micro workflow is composed of actions that are triggered by the actor who is performing the macro activity. These micro actions are typically transactional, synchronous, and short-running. In a Business Integrator solution, a micro workflow can be defined for any activity in the macro workflow graphs. It is important to note that a micro workflow, described here, is not the same as a microflow. See "Chapter 2. Business Flow Manager high-level design" on page 3 for more information on microflows.

Micro workflow is implemented by using exactly the same technology as the adaptive document controllers. There is an activity controller that implements the micro workflow associated with each activity. This activity controller is designed by using a state machine and the command design pattern, the same way that an adaptive document controller is defined.

Typically, an adaptive document controller initiates the macro workflow as a result of a state transition. When an activity becomes available in the macro workflow, Process Broker Services launches an activity controller to drive that task. When that activity is completed, the activity controller ceases to exist and the process moves to the next step or steps. When there are no more activities, the process completes.

Thus, the adaptive document and activity controllers together drive the client interaction. Multiple activities can be associated with an adaptive document instance. In the RFQ example, while sellers are in the process of responding to an RFQ, there will be multiple activity controllers associated with the same RFQ adaptive document, with each activity controller corresponding to a seller process. The macro workflow state and the micro workflow state together represent the process state. Ideally, the adaptive document state is parallel to the process state. The power of the Business Integrator programming model derives from this ability to model this parallel state space.

RFQ Process macro workflow

Using the RFQ process in the PTX as an example, the RFQ submit action by the Buyer necessitates collaboration between the Buyer-Approver, Buyer, and the Seller organizations. The macro workflow graph for the RFQ process is given in Figure 15 on page 38.

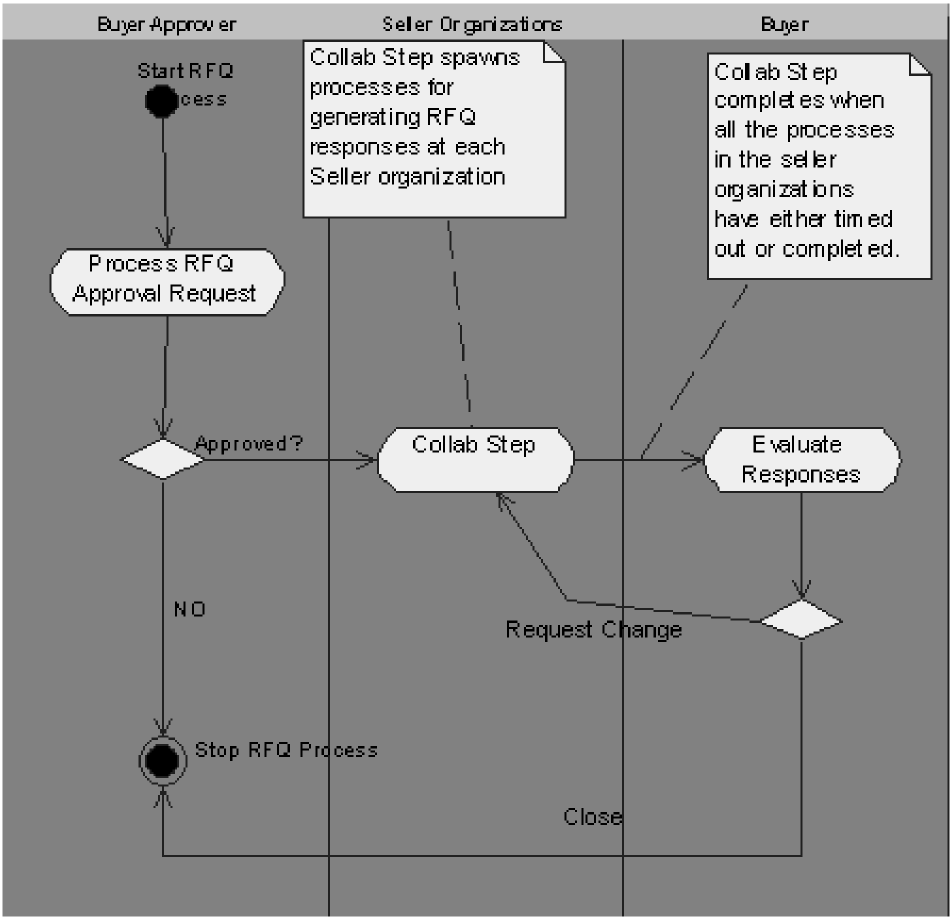
Buyer Approver
Seller Organizations
Buyer
Start RFQ Process
Collab Step spawns processes for generating RFQ responses at each Seller organization
Collab Step completes when all the processes in the seller organizations have either timed out or completed.
Process RFQ Approval Request
Approved?
Collab Step
Evaluate Responses
NO
Request Change
Stop RFQ Process
Close

Figure 15. Macro workflow for RFQ processing

In the RFQ Process, the macro workflow includes the following activities:

- RFQ Approval Request — See "RFQ Approval Request activity" for details.
- Collab Step — See "Collab Step activity" on page 39 for details.
- Evaluate Responses — See "Evaluate Responses activity" on page 40 for details.

Each of these activities is defined by a micro workflow and implemented by an activity controller.

RFQ Approval Request activity

The RFQ Approval Request activity is performed by the Buyer-Approver as the first activity in the RFQ Process macro workflow. After an approval request becomes available, the Buyer-Approver must claim the action and then decide whether the request is approved or rejected, as seen in Figure 16 on page 39 on page 39. After this decision is made, the RFQ Approval Request activity is complete and the adaptive document controller moves on to the next step in the RFQ Process macro workflow; the activity controller associated with this activity ceases to exist. If the approval request is rejected, the RFQ Process stops.

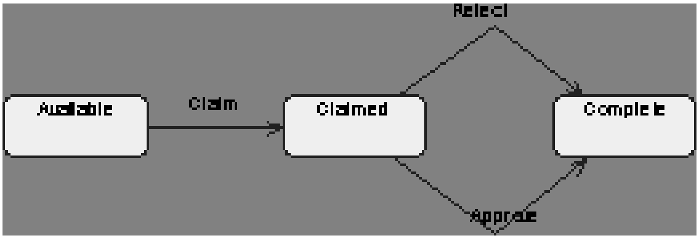

Figure 16. RFQ Approval Request activity

Table 6. State machine behavior for RFQ Approval Request activity

From	To	Event	Commands
Available	Claimed	Claim	WFActivityClaim
Claimed	Complete	Approve	• WFActivityComplete • CreateVendorList
Claimed	Complete	Reject	• WFActivityComplete • DeleteRFQData • CreateNotificationInfo • Notify

Collab Step activity

If the approval request in the RFQ Approval Request activity is approved, the next step in the RFQ Process macro workflow is the Collab Step activity; Figure 17 on page 40 shows the micro workflow for the Collab Step activity. This step involves multiple child processes being created, one for each Seller Organization, to enable the selected vendors to respond to the RFQ by using the PTX portal. The Collab Step activity completes when all of the process in the Seller Organization have either timed-out or been completed. The Collab Step activity controller then ceases to exist and the adaptive document controller moves on to the next step in the RFQ Process macro workflow.

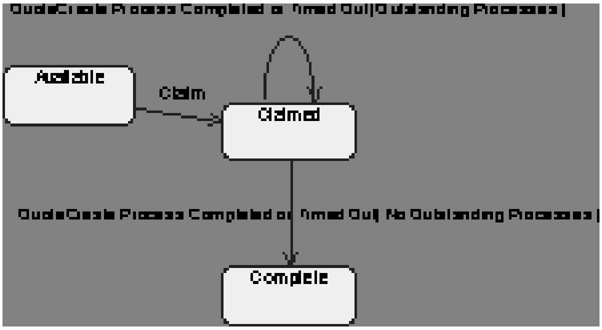
QuoteCreate Process Completed or Timed Out[Outstanding Processes]
Available
Claim
Claimed
QuoteCreate Process Completed or Timed Out[No Outstanding Processes]
Complete

Figure 17. Collab Step activity

Table 7 shows the state machine behavior:

Table 7. State machine behavior for Collab Step activity

From	To	Event	Guard	Commands
Available	Claimed	Claim		• WFActivityClaim • GenerateDynamic FanoutInfo • SpawnChild ProcessesAndTimers
Claimed	Claimed	Quote Create Process Completion or Timeout	Outstanding processes	UpdateProcess StatusInfo
Claimed	Complete	Quote Create Process Completion or Timeout	No outstanding processes	DeleteProcess StatusInfo

Evaluate Responses activity

The Evaluate Responses activity is performed by the Buyer in the RFQ Process macro workflow. After the Seller Organizations have completed the Collab Step activity, the Buyer must evaluate the quotes and either issue a change request or accept one of the quotes. If the Buyer issues a change request, the adaptive document controller sends the process back to the Collab Step activity. If the Buyer accepts one of the Seller Organization's quotes, the RFQ Process macro workflow ends. Figure 18 on page 41 shows the micro workflow for the Evaluate Responses step.

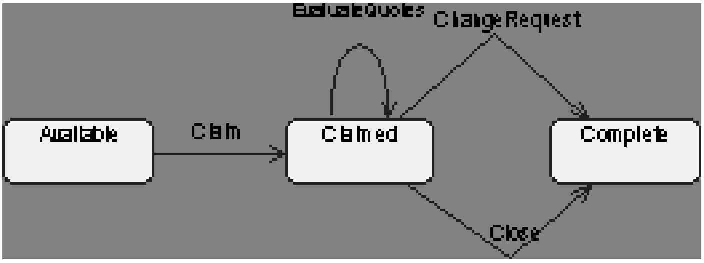

Figure 18. Evaluate Response activity

Table 8 shows the state machine behavior:

Table 8. State machine behavior for Evaluate Response activity

From	To	Event	Commands
Available	Claimed	Claim	WFActivityClaim
Claimed	Claimed	Evaluate	EvaluateQuotes
Claimed	Complete	RequestChange	• WFActivityComplete • UpdateVendorList
Claimed	Complete	Close	WFActivityComplete

Note that the event close is defined for the RFQ Process adaptive document controller as well as the Evaluate Responses activity controller. When a client invokes a service request with this event on an RFQ Process adaptive document in the submitted state and in the context of a Buyer evaluating the quotes, this event is sent to both controllers. The RFQ Process adaptive document controller invokes commands that notify the sellers of the decision, send a CreatePO message to the Process Broker Services, and delete the RFQ data from the database. The activity controller then completes the activity. For Process Broker Services to be able to handle the CreatePO message, Process Broker Services must be populated with the Purchase Order (PO) process, similar to the RFQ process described in this chapter.

Commands in RFQ activity controllers

Table 9 on page 42 shows the details of the commands, including the receivers. The system commands are indicated by an asterisk (*).

Table 9. Commands and receivers in activity controllers

Command	Receiver	Description
SpawnChildProcessesAndTimers*	BFMAdmin	This command spawns instances of the macro flow shown in Figure 9, one for each vendor. It also starts a timer for each process.
WFActivityClaim*	WWF Services	Claim a workflow activity.
GenerateDynamicFanoutInfo	RFQ BO	Create XML string containing information on child processes to be spawned.
UpdateProcessStatusInfo*	BFMAdmin	Update the completion status of child processes.
DeleteProcessStatusInfo*	BFMAdmin	Delete the completion status table.
WFActivityComplete*	WWF Services	Complete a workflow activity.
CreateVendorList	RFQ BO	Create a vendor list for this RFQ.
DeleteRFQData	RFQ BO	Delete RFQ data from database.
CreateNotificationInfo	RFQ BO	Generate information on whom to notify and what to notify.
Notify*	Notification BO	Send email notifications.
UpdateVendorList	RFQ BO	Update vendor list for this RFQ.
EvaluateQuotes	RFQ BO	Evaluate all quotes for this RFQ.

Quote Creation Process macro workflow

Figure 19 on page 43 shows the macro workflow for the quote creation process. In the Quote Creation Process, the macro workflow includes the following activities:

- Process RFQ — See “Process RFQ activity” on page 44 for details.

- Approve Quote — See "Approve Quote activity" on page 45 for details.

Each of these activities is defined by a micro workflow and implemented by an activity controller.

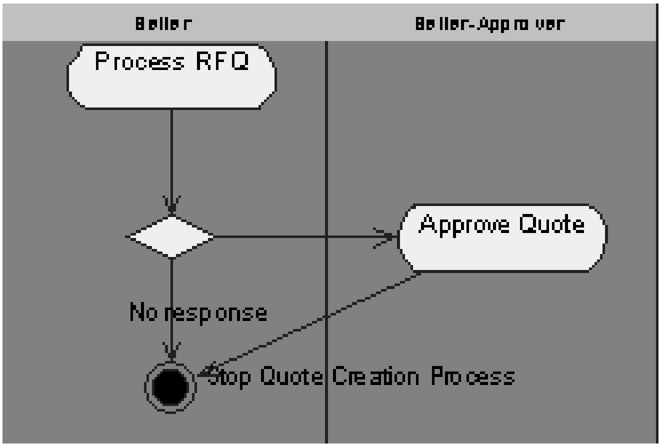

Figure 19. Macro workflow for quote creation

Several instances of this macro workflow are created, one for each vendor. It is assumed that all vendors use the PTX portal to create responses. This can create a situation where multiple activity controllers are associated with an adaptive document instance. Because a workflow activity can only be acted upon by a predefined set of actors, the client can get a handler for the appropriate activity controller by specifying the user ID and role information. Multiple clients can work with this adaptive document with each client request being handled by the appropriate activity controller.

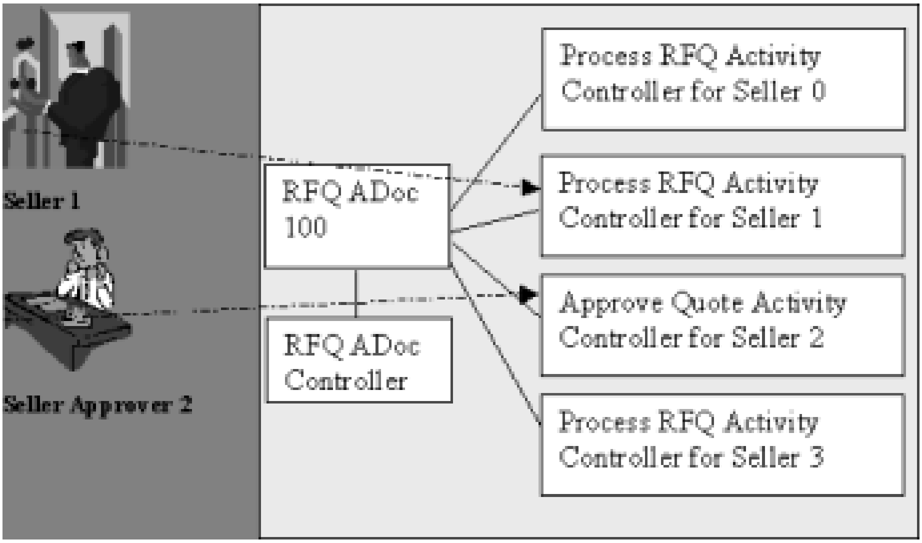
Seller 1
Seller Approver 2
RFQ ADoc 100
RFQ ADoc Controller
Process RFQ Activity Controller for Seller 0
Process RFQ Activity Controller for Seller 1
Approve Quote Activity Controller for Seller 2
Process RFQ Activity Controller for Seller 3

Figure 20. Multiple clients working with an adaptive document with multiple activity controllers

Figure 20 shows a scenario in which two seller organizations are working with an RFQ adaptive document, the first one in the Process RFQ activity while the second one is one step ahead, in the Approve Quote activity.

Process RFQ activity

Figure 21 on page 45 shows the micro workflow for the Process RFQ activity. When a request becomes available, the Seller claims the request. The Seller can then modify the quote if one already exists or create a quote if it is a new request, or the Seller can reject the request. If there is no response from the Seller or the Seller rejects the request, the Quote Creation macro workflow stops. If the Seller modifies or creates a quote, the adaptive document controller moves onto the next step and the Process RFQ activity controller associated with this Seller ceases to exist.

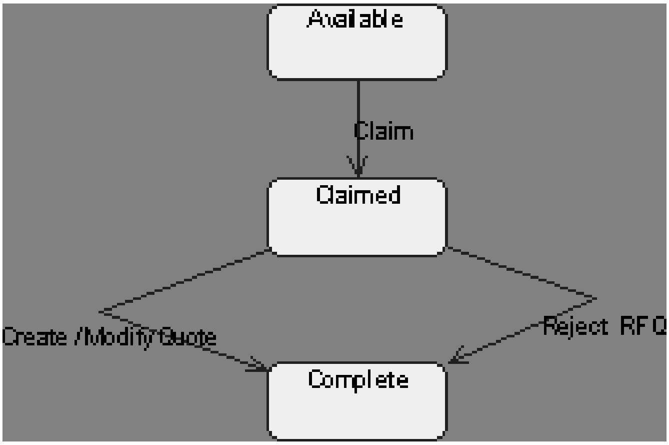

Figure 21. Process RFQ activity

Table 10 shows the state machine behavior:

Table 10. State machine behavior for the Process RFQ activity controller

From	To	Event	Commands
Available	Claimed	Claim	WFActivityClaim
Claimed	Complete	Create/Modify Quote	• Create/UpdateQuote • WFActivityComplete
Claimed	Complete	Reject	WFActivityComplete

Approve Quote activity

Figure 22 on page 46 shows the Approve Quote micro workflow. After the Seller creates or modifies a quote, the Seller-Approver must then claim the quote and either approve or reject it. Once a decision is made, the Approve Quote activity controller for this Seller-Approver ceases to exist and the adaptive document controller for the Quote Creation macro workflow stops.

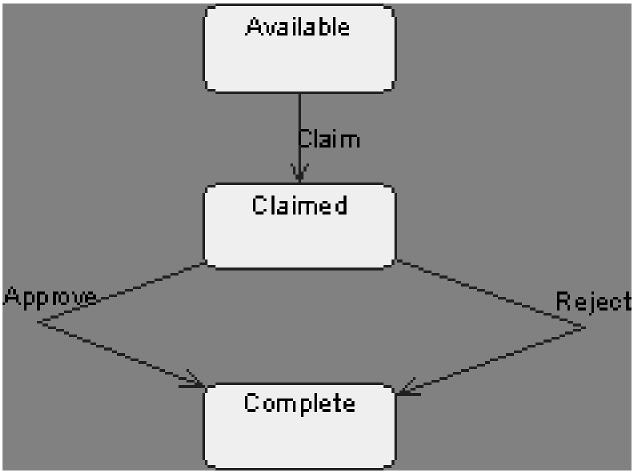

Figure 22. Approve Quote activity

Table 11 shows the state machine behavior:

Table 11. State machine behavior for the Approve Quote activity controller

From	To	Event	Commands
Available	Claimed	Claim	WFActivityClaim
Claimed	Complete	Approve	WFActivityComplete
Claimed	Complete	Reject	• RemoveQuote • WFActivityComplete

Commands in quote creation activity controllers

Table 12 shows the details of the commands and receivers. The system commands are indicated by an asterisk (*).

Table 12. Commands and receivers in activity controllers

Command	Receiver	Description
WFActivityClaim*	WWF Services	Claim a workflow activity.
WFActivityComplete*	WWF Services	Complete a workflow activity.
Create/UpdateQuote	RFQ BO	Create/update a quote.
RemoveQuote	RFQ BO	Remove a quote.

Putting it all together

Figure 23 on page 48 shows the sample complete with artifacts. All of the controllers, macro workflows, and commands are either scripted or composed graphically. The RFQ adaptive document entity enterprise bean and the RFQ Business Order enterprise bean are the only components that need to be developed, and this is done by using Visual Age for Java. All components shown shaded are part of Process Broker Services.

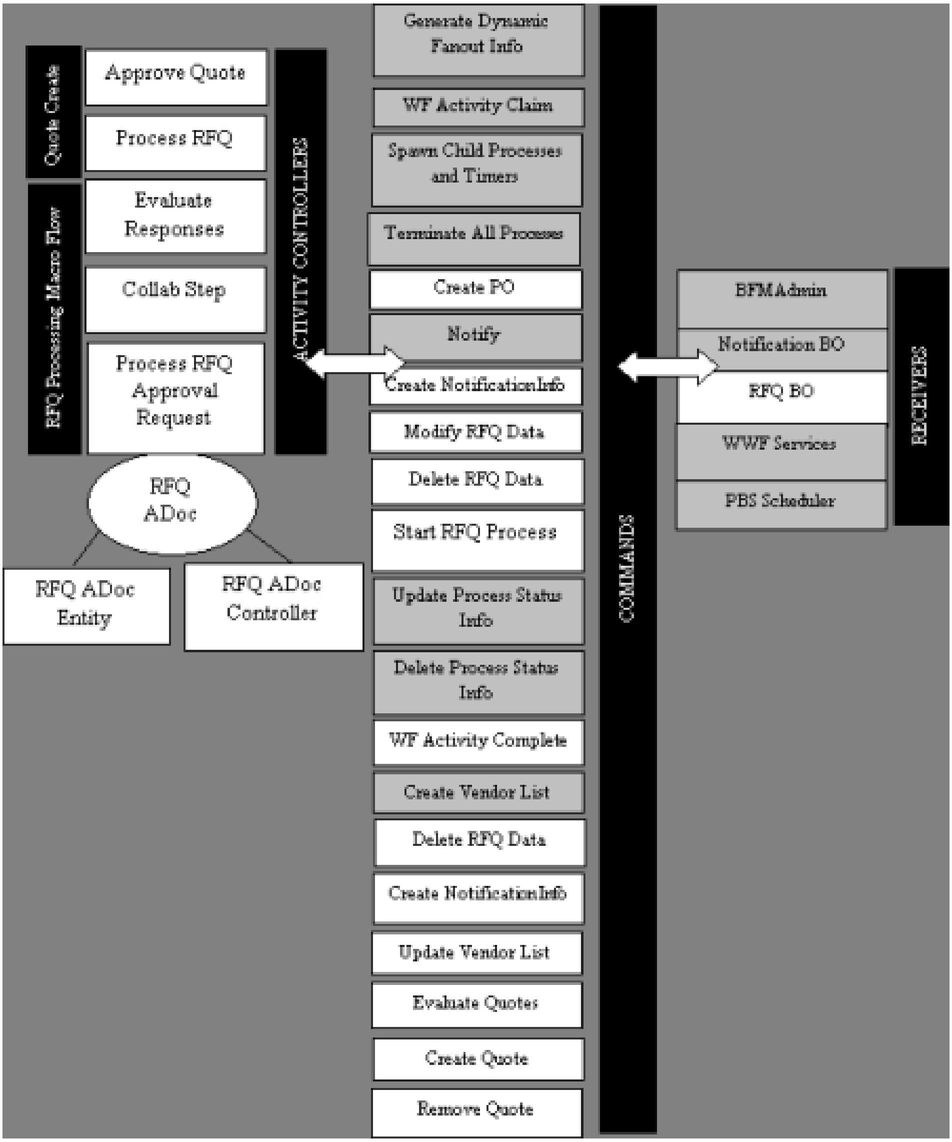

Figure 23. Solution artifacts for the RFQ sample

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

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

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION ″AS IS″ WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM United Kingdom Limited,
Intellectual Property Department
Hursley Park
Winchester SO21 2JN
United Kingdom

Such information may be available, subject to appropriate terms and conditions, including, in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained here in was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measures may have been made on development-level systems, and there is no guarantee that these measurements will be the same on generally available system. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the application data of their specific environment.

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 and cannot confirm the accuracy of performance, compatibility or any other claim related to non-IBM products. Questions on capabilities of non-IBM products should be addressed to the suppliers of those products.

Any performance data contained here in was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measures may have been made on development-level systems, and there is no guarantee that these measurements will be the same on generally available system. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the application data of their specific environment.

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 and cannot confirm the accuracy of performance, compatibility or any other claim related to non-IBM products. Questions on capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purpose of developing, using, marketing or distributing application programs conforming to IBM's application programming interfaces for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

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

- DB2
- IBM
- MQSeries
- Process Choreographer
- SecureWay
- VisualAge

- WebSphere

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

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

Rational and ClearCase are registered trademarks of Rational Software Corporation.

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

Bibliography

This bibliography lists the books in the IBM WebSphere Business Integrator and associated libraries.

IBM WebSphere Business Integrator library

The Business Integrator library consists of the following books:

- *WebSphere Business Integrator Concepts and Planning, GC34-5960*

  This book introduces the Business Integrator system, providing a high-level system overview, defining the system capabilities, and describing its value to e-businesses. This book also provides the information that you need to plan the installation of Business Integrator.
- *WebSphere Business Integrator Installation Guide for Windows NT*, *GC34-5961*

  This book is a guide to installing and configuring Business Integrator, It contains information about:
  - Selecting your required topology
  - Installing and configuring the base products and software components of Business Integrator on each machine in the topology
  - Installing and configuring firewalls and proxies
- *WebSphere Studio Business Integrator Extensions Installation Guide, SC34-5962*

  This book is a guide to installing and configuring Solution Studio, It also contains information about setting up clients and servers, and creating projects.
- *WebSphere Business Integrator Run Time*

  This book is a comprehensive guide to the Business Integrator runtime system, providing the following information:
  - Detailed conceptual information about the runtime components of Business Integrator.
  - Deployment of solutions to the runtime system
  - System administration, such as starting and stopping software components and base products, defining users, and using the Exception Console.
  - General problem determination information, including how to trace and debug, and information on obtaining help from technical support

- *WebSphere Business Integrator Messages*

  This book lists the error messages that are produced by Business Integrator and provides references to the documentation for the messages of base products.
- *WebSphere Studio Business Integrator Extensions Developer's Guide*

  This book describes how to create a Business Integrator solution, beginning with the solution design phase, to the solution implementation phase, and finally the solution deployment phase using a sample business problem. This book also provides procedures for assembling a Business Integrator solution in the run-time environment and a description of how to use the Solution Studio for solution design and implementation.
- *WebSphere Business Integrator DataInterchange for Windows NT User's Guide, SC34-5963*

  This book is a guide to installing and using DataInterchange, in the Business Integrator environment.
- *WebSphere Business Integrator Solution Samples,*

  This book discusses the two sample templates provided with Business Integrator and Solution Studio, the user-registration sample and the purchase-order management sample. It provides instructions for developing, deploying, and running the samples; it also discusses the programming model for Business Integrator Version 2.*x*.
- *WebSphere Business Integrator Process Broker Services Installation and Configuration Guide,*

  This book explains how to install and configure Process Broker Services.
- *WebSphere Business Integrator Process Broker Services Concepts Guide,*

  This book introduces the concepts involved in the Process Broker Services component of the Business Integrator system. This book also includes information on a sample that uses Process Broker Services.
- *WebSphere Business Integrator Process Broker Services Developer's Guide,*

  This book explains how to use the Application Programming Interfaces provided by Process Broker Services to create and build solution artifacts. This book also provides code samples for many of these interfaces to enable developers to understand how to implement the interfaces provided by Process Broker Services.
- *WebSphere Business Integrator Data Access Object Utility Installation and User's Reference,*

  This book describes how to install the Data Acess Object utility and explains the concept of using XML to represent SQL queries for data retrieval. This book is for solution developers who want to use XML to create database queries.

You can find the latest versions of the books at the following Web site:

```
http://www-4.ibm.com/software/webservers/btobintegrator/
```

This site contains links to the Web sites of the underlying products of IBM WebSphere Business Integrator.

Related documentation

WebSphere Business Integrator also provides a number of external application programming interfaces (API). HTML documentation that is generated using the Javadoc tool is provided for these APIs. For a list of the APIs, refer to the *WebSphere Business Integrator Run Time* book.

WebSphere Partner Agreement Manager library

The Partner Agreement Manager Version 2 Release 2 library consists of:

- *Partner Agreement Manager Installation Guide,* GC34-5964
- *Partner Agreement Manager Administrator's Guide*
- *Partner Agreement Manager User's Guide*
- *Partner Agreement Manager Adapter Developer's Guide*
- *Partner Agreement Manager Script Developer's Guide*
- *Partner Agreement Manager External API Guide*
- *Partner Agreement Manager Adapters for MQSeries User's Guide*
- *Partner Agreement Manager Channel Toolkit Configuration Guide*
- *Partner Agreement View User's Guide,* GC34-5965
- *B2B Alliance Manager iForms User's Guide,*
- *WebSphere Partner Agreement Manager Business Process Integration Adapter Guide.*

DataInterchange library

The DataInterchange Version 3 Release 1 library consists of:

- *DataInterchange Client User's Guide,* SB34-2010
- *DataInterchange Administrator's Guide,* SB34-2002
- *DataInterchange Installation Guide,* GB09-8070
- *DataInterchange Messages and Codes,* SB34-2000
- *DataInterchange Programmer's Reference,* SB34-2001

Other Libraries

You can find important information in the libraries of the following products:

- DB2® UDB
  - *IBM DB2 Universal Database Quick Beginnings Version 6.1* , S10J-8149
- MQSeries®
  - *MQSeries for Windows NT Quick Beginnings*, GC34-5389
  - *MQSeries System Administration*, SC33-1873
  - *MQSeries Using Java*, SC34-5456
  - *MQSeries MQSC Command Reference*, SC33-1369
  - *MQSeries Queue Manager Clusters*, SC34-5349
  - *MQSeries Integrator Introduction and Planning*, GC24-5599
  - *MQSeries Integrator for Windows NT Installation* , GC34-5600
  - *MQSeries Workflow Getting Started with Buildtime*, SH12-6286

- *MQSeries Workflow Getting Started with Runtime*, SH12-6287
- *MQSeries Adapter Kernel for Multiplatforms: Quick Beginnings*, GC34-5855
- *MQSeries Adapter Kernel for Multiplatforms: Problem Determination Guide*, GC34-5897
- *MQSeries Adapter Builder for Windows NT: Using the Control Center*, GC34-5882

- SecureWay®
  - *SecureWay Policy Director Up and Running*, SCT6-3KNA
  - *SecureWay Policy Director Base Administration Guide*
  - *SecureWay Firewall User's Guide*, CG31-8658
- VisualAge®
  - *VisualAge Java, Enterprise Edition Getting Started*
  - *VisualAge C++ Professional for Windows NT Getting Started*
- WebSphere™ Application Server
  - *Introduction to WebSphere Application Server*, SC09-4430

Index

File Number: BIZAAM00

Printed in U.S.A.