BEA WebLogic® Integration

Overview of WebLogic Integration

Version 10.2
Document Revised: March 2008

Contents

CHAPTER 1

Introduction

WebLogic Integration (WLI) is a unified solution for integrating business systems within an enterprise. It provides a development and run-time framework that unifies all the components of business integration – business process management, data transformation, trading partner integration, connectivity, message brokering, application monitoring, and user interaction – in a flexible, easy-to-use environment. WLI reduces the cost of management and operations by providing reliable, stable, and scalable integration solutions.

WLI combines the divergent pieces of the business integration picture – ERP, CRM, legacy applications, business users, supply chains, and trading partners – by providing a development environment that supports rapid business integration with simplified production and management.

Unified Approach to Enterprise Integration

Modern businesses operate in a diverse environment. They interact with a wide variety of clients, both within and outside the enterprise, and rely on disparate systems and processes to power their business activities. Businesses seek to integrate and extend their internal systems and processes with the goals of maximizing utilization of resources, gaining operational efficiency, and increasing revenue. Gaps exist between the business integration needs and the tools available to fulfill these needs.

Integration becomes a challenge in this kind of environment.

Regardless of your starting point – business process integration, custom application development using robust web services and controls, or development of a portal to provide employees,

partners, and customers an integrated view of applications and data – WLI provides a unified environment for building your integrated applications.

WLI provides rapid integration with WorkSpace Studio.

Figure 1-1 Rapid Integration with WorkSpace Studio

WLI equips IT staff with the means to quickly implement and bind business processes to IT resources without specialized knowledge of the deployment environment. It does this by providing access to enterprise resources such as messaging, integration controls coupled with business process modeling, human interaction workflow modeling, and data transformation.

Within the WorkSpace Studio framework, WLI supports a business process layer of abstraction and a common language for gathering requirements, validating implementation, and monitoring run-time execution. By bridging the gap between the development and integration environments, WLI helps organizations avoid accumulation of proprietary integration technologies, makes the integration effort easier, and saves money.

WLI optimizes enterprise integration by recognizing and reflecting the following design principles:

- **Loosely-coupled integrations** are easier to maintain than traditional tight and rigid integrations.
- **Asynchronous communication** is critical for conducting business operations and protecting information when the communication link becomes unavailable.

  Synchronous communication is necessary for straight-through processes or steps within an asynchronous process, that require quick response time and no persistence.

  WLI supports both synchronous and asynchronous communication with external systems.
- **Coarse-grained communication** is the key to maximize the efficiency of typically high-cost communication between loosely coupled systems.

With a common environment that recognizes that applications require integration to communicate, WLI enables reuse of technical skills across the entire lifecycle of building, integrating, and deploying applications.

Benefits of a Common Application Framework

WLI provides a robust set of general purpose tools; but your integration solution may require some custom behavior. You may, for example, want to do the following:

- Write application logic or expose processes to users for tasks such as business approvals.
- Customize the user interface to support business approval workflows.
- Build a web-based application on top of your integration application to provide access to end users.

The common application framework of WorkSpace Studio allows you to develop all of these components in a single environment. In the same IDE, you can do the following:

- Define business processes.
- Use web services.
- Access the J2EE layer for specialized application logic.
- Use NetUI and Portal resources to allow user interaction with the business process.

After you build your integration application, you can build your user interface within the IDE. You can use the JSP editor to create forms for data entry and use page groups to enable the flow of information across multiple web pages. You can host the UI on a portal and customize the user experience.

Table 1-1 summarizes the benefits of having a common application framework.

Table 1-1 Benefits of a Common Application Framework

Feature	Benefit
Industry-Standard IDE	• Leverages the power of the extended Eclipse community for skills, ideas, discussion groups, and plug-ins. • Improves developer collaboration within and across projects. • Takes advantage of prevalent Eclipse familiarity to quickly bring new team members on board.
Unified programming model	• Supports event-driven programming model based on procedural logic development. • Provides a control-based environment for defining processes and abstracting resources. • Abstracts the low-level technical details of the J2EE APIs and back-end resources.
Common look-and-feel	• Provides a consistent developer experience across different BEA WebLogic products: WorkSpace Studio, WLI, and WebLogic Portal (WLP).
Annotated Java code model	• Enables you to specify behavior and focus on handling events and calling methods, instead of writing complex code. • Provides metadata-driven application development. • Supports annotations based on the JSR 175 standard.
Web services	• Supports natively built, extensible, and integrated web services at the enterprise level. • Exposes processes as web services and invokes internal and third-party web services from IDE components. • Enables implemented processes to be automatically accessible as web services. • Follows web services standards such as Simple Object Access Protocol (SOAP) and Web Services Description Language (WSDL). • Supports access through XML messages.
Common project and deployment model	• Provides application encapsulation through J2EE mechanisms – WAR and EAR files.

Table 1-1 Benefits of a Common Application Framework (Continued)

Feature	Benefit
Controls	
Simple visual components	• Provides an easy-to-use interface. • Allows you to define the behavior of business processes through methods, events, and properties.
Extensible architecture	• Ensures that application artifacts that are built in WorkSpace Studio become controls that can be reused anywhere within the environment. • Enables developers and ISVs to develop custom controls.
Consistent mechanism for representing resources	• Abstracts resource-specific details; all resources look the same. • Reduces the learning curve for developers.
Composition	• Supports invoking controls from other controls.
Java component with Java methods	• Provides easy access to J2EE resources.
Complete access to J2EE API	• Enables J2EE developers to build the logic at the J2EE layer, package it as a control, and make it available to the application developer or integration specialist.

Integration with WebLogic Platform

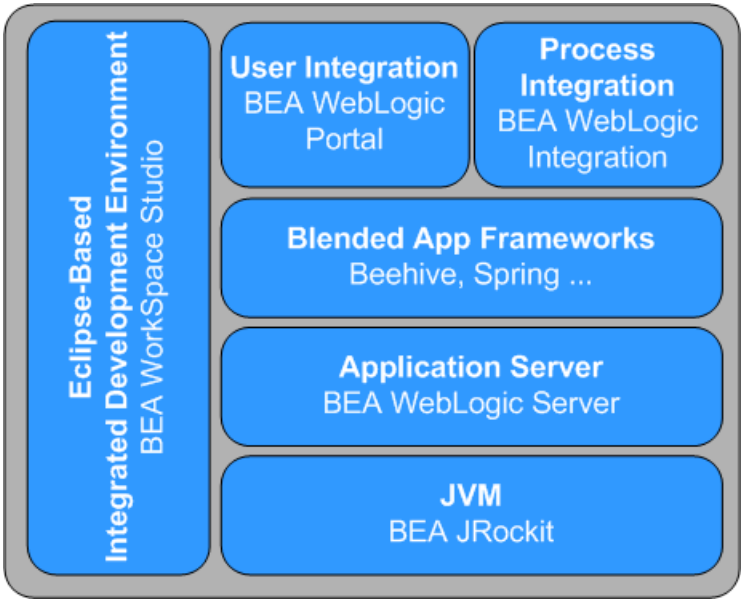
WLI is part of WebLogic Platform, which helps in streamlining business processes, developing new applications, creating web services, integrating them with existing systems, and extending e-business infrastructure through web portals.

WLI provides a single environment for:

- Integrating business processes.
- Developing custom applications by using robust web services and controls.
- Developing a portal to provide employees, partners, and customers with an integrated view of applications and data.

Figure 1-2 shows the components of the WebLogic platform that you can use independently, or in combination, as required for your application.

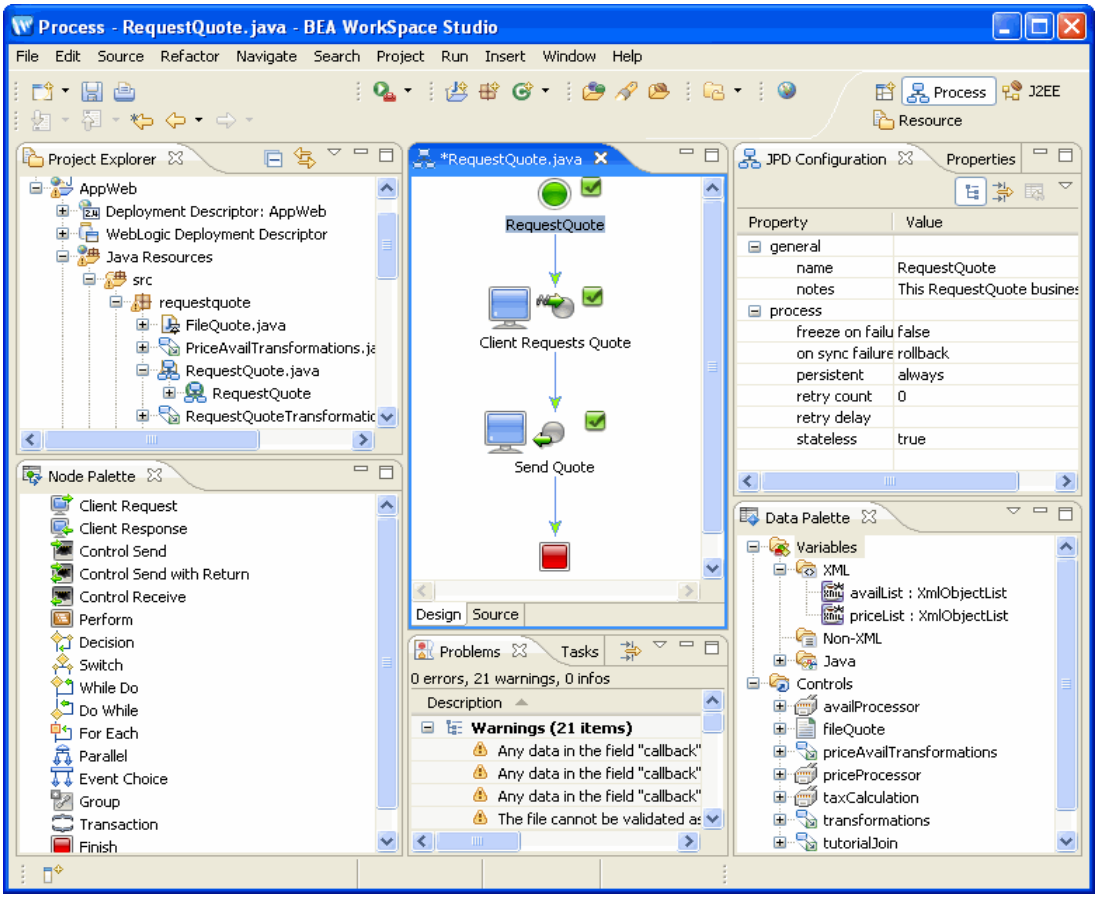
Figure 1-2 WebLogic Platform



- WebLogic Server (WLS) provides the critical infrastructure required for developing integrated solutions including security, transaction management, fault tolerance, persistence, and clustering.
- WLI leverages WLS and uses web services to integrate distributed systems within and outside the organization. It uses WorkSpace Studio to simplify application development using an Integrated Development Environment (IDE).

  WLI works seamlessly with WorkSpace Studio to provide a robust set of tools for developing and extending integration applications. It provides graphical tools for creating and changing business processes and user interaction task plans, as shown in Figure 1-3.

Figure 1-3 WLI Business Process in WorkSpace Studio



The next section outlines the key features of WLI.

Features of WLI

Table 1-2 lists the features of WLI:

Table 1-2 Features of WLI

Feature	Description
Process Integration	WLI enables you to model and integrate business processes.
Data Transformation	WLI supports transformation of data for any combination of data formats – structured XML, non-XML, or Java – by using XQuery and XSLT. Transformation is supported for incoming data, outgoing data, as well as data within a process. Data transformation can be implemented in a graphical design view. Developers can view and change the underlying code in the source view. They can also test the transformation in a test view. Complex transformations such as joins, unions, typeswitch, if, and FLWOR (For, Let, Where, Order by, Return) operations can be designed quickly by a simple drag-and-drop mapping.
Interaction with External Systems	WLI provides the following features to help you design applications that can interact efficiently with external systems: • Message Broker: The message broker feature provides rules-based message routing by using a channels-based publish-subscribe broker to transport events in a loosely coupled manner. This enables high-performance, low-latency message routing between applications. • Event Generators: Event generators publish messages to Message Broker channels in response to system events. • Controls: Controls enable nonexperts to achieve rapid integration by dragging and dropping simple component interfaces that represent the resources being integrated. Over a dozen prebuilt controls are available out-of-the-box for access to database, file, HTTP, messaging, service broker resources, and for human interaction. The controls container is based on Apache Beehive.
Human User Integration	WLI includes a full-featured worklist system to manage end-user interaction for process exceptions, approvals, and status tracking. You can create a reusable sequence of end-user steps, which can be used with one or many processes by using a drag-and-drop design interface and out-of-the-box portlets; you can also generate automated forms. The worklist feature includes centralized user and group management, and user rules and authorization for secure participation within processes.

Table 1-2 Features of WLI (Continued)

Feature	Description
Trading Partner Integration	WLI enables rapid and secure online connection with suppliers and customers through leading standards such as RosettaNet and ebXML. These protocols provide: • Secure messaging, digital signatures, and encryption. • Recoverable and trackable messages. • Dynamic configuration updates. WLI accommodates a full range of partners — from full hub-to-hub interaction to zero-weight client access through portal, browser, or FTP access. You can manage trading partner profiles with streamlined import and export of configurations.
Integration with Back-End Systems	WLI provides access from Workshop controls to pre-built BEA WebLogic Adapters and custom adapters.
Administration and Management	WLI includes a portlets-based administration console, which facilitates integration-focused lifecycle management of running processes, deployed applications, message broker traffic, trading partner activity and parameters, and worklists. It gives administrators full and secure visibility into the distributed integration environment. The UI is easy to use and enables users to navigate quickly across the modules of the console.
BPEL Import and Export	WLI supports import and export of BPEL 1.0 and 2.0 processes.
Eclipse-Based IDE	WorkSpace Studio contains an open source IDE framework, which follows the best practices of Eclipse 3.2.2 and works with Eclipse plug-ins. WLI uses Eclipse concepts such as Workspace, Workbench, Editors, Views, Resources, Projects, Perspectives, and Facets.

Table 1-2 Features of WLI (Continued)

Feature	Description
Integration with AquaLogic Service Bus	Integration of WLI and ALSB provides a cost-effective solution for building, connecting, and managing integrated process-driven services within and outside the enterprise, by combining the power and flexibility of WLI with the high-performance, stateless mediation of ALSB. This integration provides the following benefits: • You can install WLI and ALSB in the same *BEA_Home*, and you can deploy WLI and ALSB applications in either a single domain (with a unified run time) or in separate domains. • Developers can navigate easily between ALSB and WLI design perspectives. • Security and transaction contexts are propagated seamlessly from ALSB to WLI and vice versa. For more information, see the AquaLogic Integrator documentation at http://e-docs.bea.com/alint/docs30/alintuser/. **Note:** ALSB is part of the BEA AquaLogic family of service infrastructure products; it manages the routing and transformation of messages in an enterprise system. For more information, see the ALSB documentation at http://edocs.bea.com/alsb/docs30/index.html.
Integration with AquaLogic Enterprise Repository	WLI includes a repository browser using which you can connect to ALER, search for services stored in ALER, and use them in WLI. You can also store metadata about WLI artifacts in ALER. **Note:** ALER is a SOA repository that provides the tools to manage and govern the metadata for any type of software asset, from processes and services to patterns, frameworks, applications, components, and data services. For more information, see Using ALER with WLI Applications.
Integration with AquaLogic Enterprise Security	Integration of WLI with ALES allows you to implement policy-driven security, providing increased security for application- and system-level resources. **Note:** ALES is a fine-grained entitlement management solution that combines centralized policy management with distributed policy decision-making and enforcement. This combination provides management and control of your critical applications and resources with uncompromised performance and reliability, allowing you to adapt to changing business requirements quickly and easily. For more information, see the ALES documentation at http://edocs.bea.com/ales/docs30/index.html.

CHAPTER 2

Process Integration

The process integration functionality of WLI enables you to model, test, deploy, and maintain complex business processes that integrate diverse enterprise systems, cross-enterprise applications, and end-user decision makers.

From the BPM perspective, the enterprise is a set of business services that are accessed through controls and can be orchestrated to model a business process. WLI supports synchronous and asynchronous communications, and stateless and stateful processes.

As you design business processes using the graphical tools (**Design** view), Workshop generates the source code automatically in a process file. When you need to write and edit Java code, you can switch to the **Source** view with a single click.

Table 2-1 lists the key process integration features of WLI.

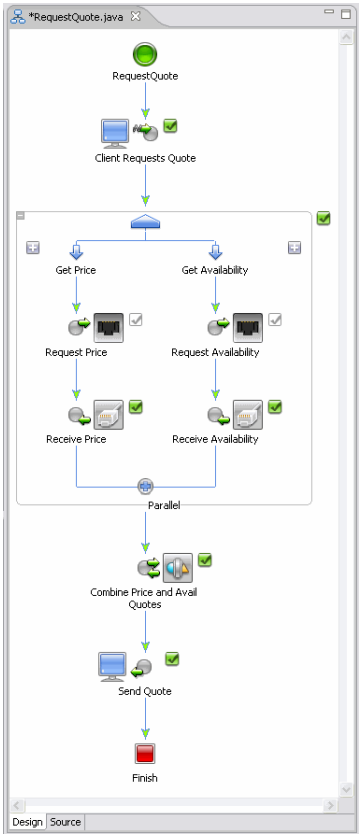
Table 2-1 Process Integration Features

Feature	Benefits
Unified access to resources through Controls	• Developers can view business activities as services and orchestrate them using a process. • Processes interact seamlessly with users, applications, back-end resources, and resources within and outside the firewall.
Simplified and structured business processes	• The flow is based on XML and the operations are Java-based.

Table 2-1 Process Integration Features (Continued)

Feature	Benefits
Graphical editing of processes for high-level integration scenarios	• Developers can model complex business logic quickly by dragging and dropping icons into a flow (see Figure 2-1). They can focus on defining the overall business logic rather than on implementation details. Processes are created as Java classes. The process files contain the metadata describing the business logic.
Support for Java code in process nodes	• Developers can, with just one click, switch from the **Design (**graphical) view to the **Source** (code) view, and then add or change Java code within process nodes.
Process implementation optimized for performance	WLI supports: • Stateless synchronous processes • Stateless asynchronous processes • Stateful asynchronous processes

Figure 2-1 Graphical Representation of Business Process



Web Services as Process Resources

WLI leverages web services, asynchronous communication, and XML messaging at the platform level. These services can be used across internal and external integrations, giving application developers the tools to simplify development and integration of loosely coupled and asynchronous applications.

WLI provides native support for web services, including security and reliable messaging. Web services can be invoked from within a WLI process, and processes can be exposed as a web service and made available as resources to other applications and application components.

Synchronous and Asynchronous Processes

A business process can be designed to be synchronous or asynchronous based on the method that is used to invoke the process.

- A synchronous process is invoked by a synchronous method; it returns a response to the client after the process is executed.

  Note: A synchronous process can also contain asynchronous operations, but these must be added after the starting event in the process flow, so that, at run time, the asynchronous processes are executed after the synchronous starting event is completed.

- An asynchronous process is invoked by an asynchronous method. The starting event in such a process is represented by an asynchronous node. An asynchronous process can call synchronous or asynchronous methods without additional configuration.

You can also enable synchronous clients to interact with processes that have asynchronous interactions with resources. A synchronous Workshop client such as a JSP or portal page that uses a Java control might, for example, need to invoke a process and then wait for a response (block). While the client is blocked, the process may perform asynchronous activities such as queueing a JMS message and waiting for a JMS receive, and then return the response to the client, after which the client is unblocked.

Stateless and Stateful Processes

Processes can either be stateless or stateful.

- Stateless processes provide high performance because there is no overhead for maintaining the state in a database during the transaction.

- For a stateful process, the state of the process is persisted in a database. Stateful processes are used when data reliability and recovery are essential. However, persisting the state could affect performance of the process. Nonpersistent stateful processes can be defined at design time to minimize the effect on performance.

When you set the properties of the process, the following types of persistence can be defined:

- **Always**: The state is always persisted in a database.

- **Never**: The state is stored only in memory (and never in a database) and lasts only for the duration of the current session.

- **On overflow**: The state is persisted only after a number (specified in the Max Bean in Cache deployment descriptor file) is reached.

Process Calls

Processes can expose their functions to clients in several ways, including through WSDL files, process controls, service broker controls, and JPD proxies.

Process controls can be used for invoking other processes over an optimized transport. The service broker control must be used for invoking other processes and WebLogic Workshop 8.1-based web services over HTTP or JMS transport. The invoked processes and web services may be deployed on the same domain or on a different domain.

Any Java client – including standalone Java applications, EJBs, JSPs, and servlets – can communicate with any process by using JPD proxies. When the clients use JPD proxies, they make Java method calls using Remote Method Invocation (RMI).

For more information about creating processes, see *Guide to Building Business Processes*.

CHAPTER 3

Data Transformation

Data transformation enables you to translate between XML, non-XML, and Java data formats, allowing you to integrate heterogeneous applications rapidly. Data transformations in WLI can be packaged as controls and reused in multiple business processes and applications.

In WLI, XML data can be transformed by using either XQuery expressions or eXtensible Stylesheet Language Transformations (XSLTs).

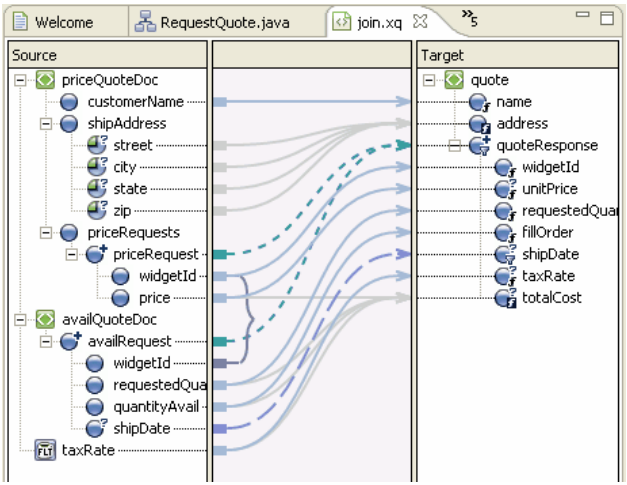
XQuery is a standards-based query language with the simplicity of SQL-like expressions. WLI provides an easy-to-use graphical mapping tool called XQuery Mapper, which enables you to generate complex transformations easily.

WLI supports data transformation for the following versions of XQuery:

- XQuery 2004: Graphical design view (XQuery Mapper), source code view, and test view.
- XQuery 2002: Source code and test views.

Figure 3-1 shows the XQuery Mapper.

Figure 3-1 XQuery Mapper



The mapper enables conversion of data of different types. For example, XML data that is valid according to one XML schema can be transformed to an XML document that is valid according to a different XML schema.

Table 3-1 lists the main features of data transformation.

Table 3-1 Data Transformation Features

Feature	Benefits
Data transformation	• Transformations are packaged as controls, which can be treated as resources and reused across multiple processes and integration solutions. • Transformation of data can be performed between any of the following input-output data types: XML data, non-XML data, Java primitives, and Java classes. • Transformations can have multiple data sources. • WLI enables transformation of XML grammar.
Integration with business processes	• WLI enables transformation of data in a business process using XQuery and XSLT. • Data that is received as an incoming message to the process can be transformed. • Data can be transformed before the process sends an outgoing message. • Data can be transformed within the process.

Table 3-1 Data Transformation Features (Continued)

Feature	Benefits
XQuery mapper	• WLI provides a graphical tool for data transformation between any combination of XML, non-XML, and Java data formats. • The XQuery mapper enables transformation through a simple drag-and-drop mechanism. • WLI supports complex operations, including joins, unions, typeswitch, if, and FLWOR (For, Let, Where, Order by, Return) expressions.
Format builder	• This tool enables creation of metadata to describe non-XML data. For more information, see the Format Builder documentation.

For more information about data transformation using the XQuery Mapper, see *Transforming Data Using the XQuery Mapper*.

CHAPTER 4

Message Broker

Message brokers provide business processes a channel-based publish and subscribe communication mechanism, which enables processes to communicate in a loosely coupled, anonymous mode.

A purchase order routing process can, for instance, subscribe to a message broker channel called NewOrderEntered; every time a new order message is published to the NewOrderEntered channel, the purchase order routing process is activated.

You can specify the channels to which a process publishes and subscribes. Publishers can broadcast messages on the channels and consumers such as processes and other back end resources can subscribe to them. In this way, the message broker facilitates a loosely coupled interface. You can add new publishers and new subscribers at run time.

Message brokers support event generators that can publish events from external sources to the message broker channels.

Note: For more information about event generators, see Chapter 5, "Event Generators."

Table 4-1 lists the message broker features of WLI:

Table 4-1 Message Broker Features

Feature	Benefits
Publish and Subscribe	• The publish and subscribe communication mechanism provides high-performance throughput. • Processes can subscribe dynamically to message broker channels through controls. Static subscriptions can be specified at the start node of the process. In this case, the process starts when it receives a message from a channel to which it is subscribed. • Subscriptions can start a new process or be used within a running process to block a message. • XQuery filters can be used to refine message selection on subscription.
Message Broker Channels	• WLI supports static and dynamic binding to channels. • Channel typing explicitly defines the structure of the message that the channel can route. • The channel-naming hierarchy is defined. • Event generators publish messages to channels from external sources. • Event generators can be bound dynamically to channels at run time using the WLI Administration Console. • WebLogic adapters can publish events to channels.
Message Routing	• WLI supports routing from event generators to stateless processes. • Message routing takes place as a single transaction; there is no need to save the state.
Rules	• Business process logic defines the trigger, transformation, and routing rules. • The business logic can be defined in the form of rules with short execution times rather than as time-consuming processes. • Stateless processes implement rules.

Event Generators

Event generators trigger events in response to activities that occur in the systems associated with them. Each event generator is associated with a message broker channel. When a system event occurs, the event generator publishes information about the event through the relevant message broker channel. Channel rules can be configured for each event generator.

Table 5-1 lists the types of event generators that are supported by WLI.

Table 5-1 Event Generator Types

Event Generator Type	Description
File	A file event generator polls for files or directories on a local drive or remote server (FTP or SFTP) and generates an event when an operation – read, write, or append – is performed on the file. In addition, file manipulation operations such as copy, rename, and delete can be monitored. The monitored files can contain XML objects, raw data (binary), or strings. Details of the file operations are published on message broker channels as XML or binary objects.
Email	This type of event generator polls e-mail accounts for messages and publishes the content to message broker channels.
JMS	A JMS event generator polls JMS queues and topics for messages. These messages are then published using the message broker channels.

Table 5-1 Event Generator Types (Continued)

Event Generator Type	Description
Timer	A timer event generator creates events at user-specified times and publishes the events to message broker channels. When the timer event generator detects that a specified time has passed, it publishes a message to a message broker channel. The content of the message can be specified in the channel rules defined for the event generator. These are always XML messages. Timer event generators can be associated with business calendars that are defined in the Worklist Console. For more information, see: • About Business Calendars and Business Time Calculations in Business Calendar Configuration. • Defining Channel Rules for a Timer Event Generator in Event Generators.
RDBMS	An RDBMS event generator can be used to generate an event when an insert, update, or delete action takes place on any database table associated with the event generator. The RDBMS event generator can also be used for querying data available in a database. The events generated by the RDBMS event generator are published on message broker channels. For more information, see *RDBMS Event Generator User Guide*.
HTTP	The HTTP event generator is a servlet, which takes HTTP requests, checks for the content type, and then publishes the messages to message broker channels. The HTTP event generator supports synchronous subscription; it also supports a multipart response.

Table 5-1 Event Generator Types (Continued)

Event Generator Type	Description
MQ Series	MQ Series is the messaging service queue provided by IBM. An MQ Series event generator polls the MQ Series message queue and publishes the messages through message brokers channels. WLI provides and supports integration of the IBM Websphere® MQ Series messaging service queue through the following options: • WLI JMS control. For more information, see WLI JMS Control in *Using Integration Controls*. • JMS event generators in the WLI Administration Console. For more information, see Event Generators in *Using the WebLogic Integration Administration Console*. • Async sample illustrating the use of JMS with MQ Series. For more information, see Async Binary Update Sample. • WLI MQ Series control. For more information, see MQ Series Control in *Using Integration Controls*. • BEA Adapter for MQ Series For more information, see the BEA WebLogic Adapter for MQ Series 8.1 Documentation. **Note:** The WLI controls and adapters use the MQ Series APIs. To resolve issues in an MQ Series control or adapter that is not part of WLI or is specific to IBM WebSphere MQ, the APIs might require workarounds. JMS-based MQ integration, however, does not have the same support-related limitations as IBM WebSphere MQ APIs. For information about the BEA products for which IBM has explicitly stated its support in conjunction with WebSphere MQ, see the following URL: http://www-306.ibm.com/software/integration/wmq/requirements/index.html

Table 5-1 Event Generator Types (Continued)

Event Generator Type	Description
TIBCO RV	TIBCO Rendezvous is a messaging software provided by TIBCO. It enables exchange of data across applications running on distributed platforms. The TIBCO RV event generator listens for messages on a Rendezvous subject and raises events to the message broker when it receives the desired message. The messages are received in most formats supported by Rendezvous, converted to binary, and then published on the message broker channels. These messages can be in XML, string, and the TIBCO proprietary Rendezvous message format. **Note:** Use of the TIBCO RV control and event generator with WLI in no manner confers or grants the right to use TIBCO Rendezvous including dynamic libraries. In order to use TIBCO products, you must obtain a license from TIBCO. For more information, see http://www.tibco.com. For more information, see *TIBCO Rendezvous Event Generator User Guide*.

Event generators can be managed, deployed, and monitored using the WLI Administration Console. Event generators can be deployed on stand alone servers and on clusters.

For more information, see Event Generators in *Using the WebLogic Integration Administration Console*.

CHAPTER 6

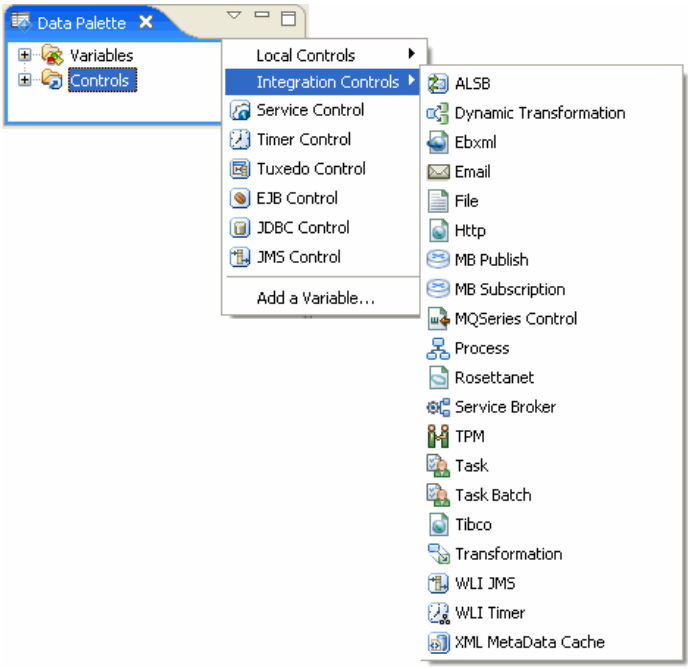
Controls

WLI provides a set of controls that you can use in integration projects to access external resources.

Note: WLI applications can access external resources through the high-level controls of Workshop as well. Workshop provides a consistent mechanism for interacting with resources across all the components of Workshop, WLI, and WLP.

Figure 6-1 shows the WLI controls.

Figure 6-1 WLI Controls



WLI controls are based on the Apache Beehive open source framework (http://beehive.apache.org/). Beehive is a light-weight component framework that helps programmers encapsulate application logic and leverage metadata annotations into their programming model. Developers can create custom controls or use the prebuilt system controls.

Note: WLI controls are available in Workshop only if you have a license for WLI.

The following sections describe the controls delivered with WLI.

ALSB Control

WLI processes can access AquaLogic Service Bus (ALSB) proxies by using generic Workshop controls such as web service control, JMS control, and HTTP control. The ALSB control serves as an ALSB proxy-specific control that ensures efficient communication. It can be used regardless of whether WLI and ALSB are on the same or separate servers. The control exposes an ALSB proxy and its metadata to WLI processes.

The ALSB control operates in two modes:

- Loosely coupled: WLI and ALSB in separate domains (default).
- Local: WLI and ALSB on a single server.

Dynamic Transformation Control

The dynamic transformation control provides the ability to select, at run time, the transformation that must be invoked. The format of the input and output data need not be specified while designing the process. Instead, the process identifies the input data and transforms it to the desired output format at run time.

The dynamic transformation control uses transformations that are already defined and tested, such as those created with the transformation control. This control has base methods defined for various transformation types such as XML, XSLT, and MFL. Custom methods for executing different types of transformations can also be created. At run time, the transformation type is identified and executed using these methods.

ebXML Control

The ebXML (e-business XML) protocol is a modular suite of specifications that enables enterprises of any size and in any geographical location to conduct business over the Internet. It is sponsored by UN/CEFACT and OASIS. For more information about ebXML, see http://www.ebXML.org.

The ebXML control enables processes to exchange business messages and data with trading partners using the ebXML protocol. This control supports both ebXML 1.0 and ebXML 2.0 messaging services.

Email Control

This control enables processes to send e-mails to a specific destination.

Note: To receive e-mail, you must use the Email event generator, which then publishes incoming e-mails to message broker channels.

File Control

The file control enables processes to read, write, or append to files that are either in the local file system or on remote systems (FTP or SFTP server). In addition, the file control supports file

manipulation operations such as copy, rename, and delete. You can also retrieve a list of the files stored in a specific directory. The files can be one of the following types: XML, binary, and string.

The methods available to a file control depend on the type of data in the file. For a string or XML type file, you can specify the character set encoding. While processing a file, you can specify the delimiter for the file as the record size. If no delimiter is specified, the file is processed one line at a time. In the case of string type files, you can also specify the number of bytes or any character as a delimiter.

HTTP Control

The HTTP control is used to provide outgoing HTTP access to Workshop clients. This control can be used to work with HTTP requests and process responses. The HTTP control supports the Get and Post request methods for data transfer. By using Get, you can send business data along with the URL. By using Post, you can send large documents (binary, XML, or string) to the server within the body of the request. You can specify HTTP control properties in an annotation or pass dynamic properties through an XML variable. Using this control, you can send an HTTP or HTTPS request to a URL and receive specific HTTP response header and body data.

Message Broker (MB) Publish and Subscription Controls

The message broker resource provides a publish-subscribe, message-based communication model for WLI processes, and includes a powerful message filtering capability.

The **MB Publish** control is used for publishing messages to message broker channels. You bind a message broker channel to the MB publish control when you declare the control, but the binding can be overridden dynamically. You can add additional methods to your extension (subclass) of the MB publish control.

The **MB Subscription** control is used to dynamically subscribe to channels and receive messages. When you create an instance of the control for a process, you bind the channel and, optionally, an XQuery expression for filtering messages. The bindings cannot be overridden dynamically. The MB subscription control interface includes methods that allow your process to subscribe to and unsubscribe from the bound message broker channel.

In addition to the dynamic subscriptions that you design at control nodes in your process, you can design static subscriptions at the start nodes to receive messages from message broker channels. In other words, a process can be initiated by subscribing (synchronously or asynchronously) to a message broker channel and started through an event.

MQ Series Control

MQ Series is a messaging service queue provided by IBM, which enables message transfer between applications. The sending application puts a message in a queue and the receiving application gets the message from the queue.

The MQ Series control enables WLI processes to send and receive messages using MQ Series queues. Using the MQ Series control, you can send and receive binary, XML, and string messages. You can specify MQ Series control properties while configuring the MQ Series control or dynamically at run time. By default, the MQ Series control handles transactions implicitly for each Put and Get method individually, without having to set an explicit transaction boundary. Transaction boundaries can, however, be set explicitly.

Using SSL, you can enable one-way authentication (server-side only) and two-way authentication (client- and server-side).

WLI provides and supports integration of the MQ Series messaging service queue (now known as WebSphere® MQ) through the following options:

- WLI JMS control.

  For more information, see WLI JMS Control in *Using Integration Controls*.

- JMS event generators in the WLI Administration Console.

  For more information, see Event Generators in *Using the WebLogic Integration Administration Console*.

- Async sample illustrating the use of JMS with MQ Series.

  For more information, see Async Binary Update Sample.

- WLI MQ Series control.

  For more information, see MQ Series Control in *Using Integration Controls*.

- BEA Adapter for MQ Series

  For more information, see *BEA WebLogic Adapter for MQ Series 8.1 Documentation*.

  Note: The WLI controls and adapters rely on use of the MQ Series APIs. To resolve issues in an MQ Series control or adapter that is not part of WLI or is specific to IBM WebSphere MQ, the APIs might require workarounds. JMS-based MQ integration, however, does not have the same support-related limitations as IBM WebSphere MQ APIs.

For information about BEA products that IBM has explicitly stated it supports in conjunction with WebSphere MQ, see the following URL: http://www-306.ibm.com/software/integration/wmq/requirements/index.html

Process Control

A process control is used for sending a request to and receiving a callback from another process. This control is typically used to call a sub process from a parent process. Process control invocations are RMI calls.

You can create a process control by either selecting it from the list of WLI controls or generating it from the process file. When you insert the process control, you can specify the target process and the start method associated with that process. Optionally, you can use the query builder to decide, at run time, the specific subprocess that the control must call. When you generate the process control using the process file, a control extension file is created automatically for the control. You can double-click the control extension file to view the control in the design view or drag it to the data palette and use any of the methods associated with the control.

RosettaNet Control

RosettaNet is a protocol that enables enterprises to conduct business over the Internet. For more information, see http://www.rosettanet.org.

The RosettaNet control enables WLI processes to exchange business messages and data with trading partners using the RosettaNet protocol. You can use this control only in initiator processes to manage the exchange of RosettaNet business messages with participants. The RosettaNet control supports RosettaNet version 1.1 and 2.0 of the implementation framework.

For more information, see Introducing RosettaNet Solutions in *Introducing Trading Partner Integration*.

Service Broker Control

The service broker control allows processes to send requests to and receive callbacks from other process or Web Service Description Language (WSDL) files.

The service broker control does not support the latest version of web services standards such as WS-Addressing, WS-Security, and SOAP 1.2. To access web services that support these standards, use the web service control delivered with Workshop.

Task and Task Batch Controls

The task and task batch controls are worklist controls that enable you to introduce user-assigned tasks and task management to WLI, so that you can build a worklist system. These worklist controls enable the automated manipulation, creation, and management of tasks.

TIBCO RV Control

Rendezvous is a messaging software provided by TIBCO. It enables exchange of data across applications running on distributed platforms. The TIBCO RV control in WLI enables seamless connection to (and transfer of data with) TIBCO Rendezvous using the Rendezvous daemon. It enables communication through many of the features provided by TIBCO Rendezvous, including certified message delivery and distributed queue. The sending and receiving applications can be on multiple platforms as long as the Rendezvous daemon is running on the host machine or is remotely accessible to the host.

Note: Use of the TIBCO RV control and event generator with WLI in no manner confers or grants the right to use TIBCO Rendezvous, including "dynamic libraries." To use TIBCO products, you must obtain a valid license from TIBCO. For more information, see http://www.tibco.com.

TPM Control

The TPM (Trading Partner Management) control enables WLI processes and web services to query (read-only access) for trading partner and service information stored in the TPM repository. Information such as the trading partner's name and business ID, default trading partner, basic and extended trading partner properties, default bindings (ebXML or RosettaNet), services, service profiles, and service profile bindings (ebXML, RosettaNet, or web service) can be queried and retrieved from the TPM repository. However, only active profile services and active trading partners can access this repository.

Note: Since access to the repository is read-only, you cannot change trading partner and service information. These details can be changed only by using WLI Administration Console.

WLI JMS Control

Java Message Service (JMS) is a Java API for communicating with messaging systems. The WLI JMS control enables processes to easily interact with messaging systems that provide a JMS implementation.

Each WLI JMS control is associated with a specific facility of the messaging system. Once a WLI JMS control is defined, processes can use it like any other Workshop control.

XML MetaData Cache Control

This control is used by processes to access and retrieve XML metadata maintained in the XML metadata cache. This cache is managed by WLI Administration Console or the MBean API, which allows users to create their own NetUI-based consoles.

The XML metadata cache is a global, domain-wide cache. Data that is maintained in the cache can be accessed by any process that is deployed in that domain. The cache can also be used for sharing data within a cluster. The cache is used primarily for maintaining configuration metadata. Data is stored in the cache as key-value pairs where the key is of type String and the value contains XML data. Data from the cache is made available permanently through file-based storage. For each XML document that is added to the cache, a new XML metadata cache file is created. The XML metadata cache control in a process uses the key to retrieve XML metadata associated with the key value from the cache.

For more information about WLI controls, see *Using Integration Controls*.

CHAPTER 7

Human User Integration

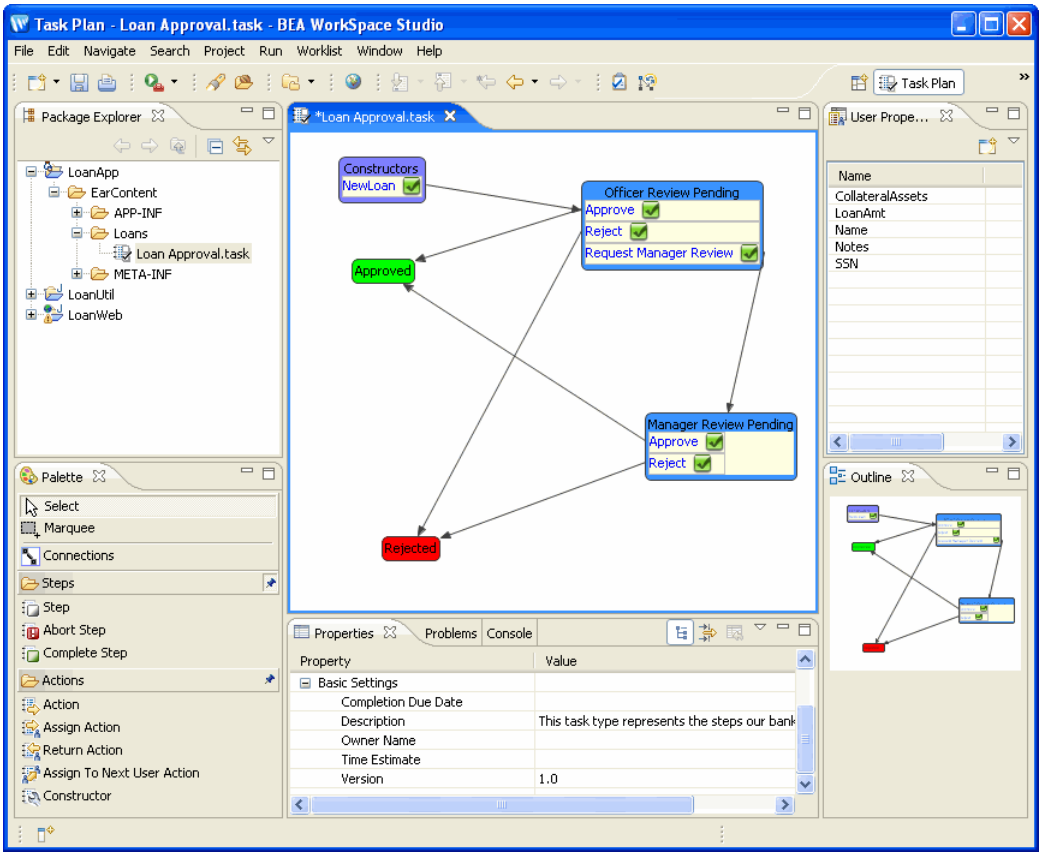
WLI supports integration of business processes with human users through worklists.

Worklists provide capabilities to manage users, groups, and roles, and to manage the routing of tasks to people within an enterprise. They enable people to collaborate in business processes; that is assign tasks, track the status of tasks, handle approvals, and so on. Tasks such as receiving documents, and approving, changing, and routing them occur in most business processes.

In worklists, a task plan is the central element. Designers can string together multiple steps to form the task plan. Each step in a task plan represents a distinct phase toward completion of a specific business goal, such as approving a loan request. As business users work on each step, they can choose actions that will take them to the appropriate next step based on the rules and conditions defined in the task plan.

WLI provides a drag-and-drop design framework to define task plans. The design framework is delivered as a perspective within WorkSpace Studio, as shown in Figure 7-1.

Figure 7-1 Task Plan Perspective in Worklist Applications

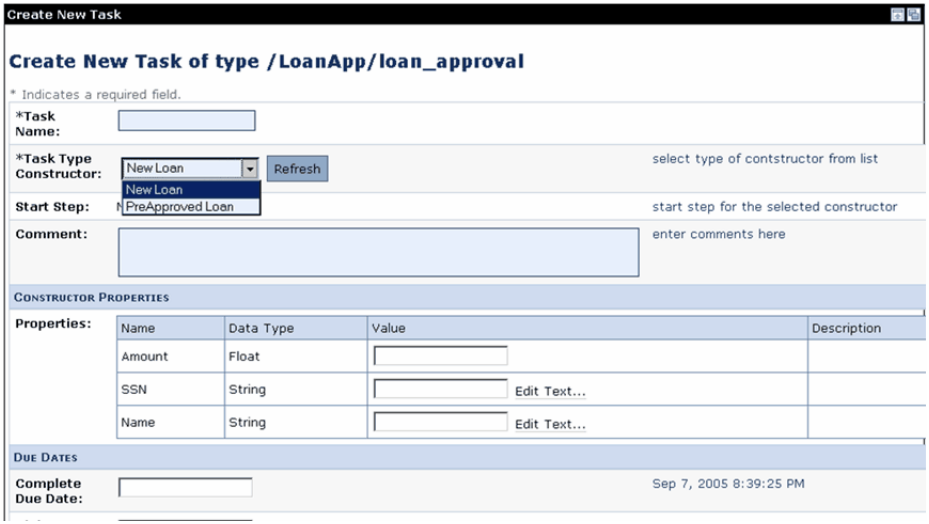


Processes can deal with task instances through controls and the message broker. The task control can be used to interact with a single task instance in a process. The task batch control can be used to interact with multiple task instances, queries, and bulk operations. The worklist events related to any task instance are published to a message broker channel. This allows processes to subscribe to and filter specific events and act on them, enabling loosely coupled integration between the worklist and the processes.

As you create your task plan, WLI automatically generates an end-user facing form that can be used to capture data from business users. These forms can be customized to suit the look and feel of your environment.

Figure 7-2 shows an example of a generated form.

Figure 7-2 Automatically Generated Form for Capturing End-User Data



WLI also contains a set of out-of-the-box portlets – user and manager – that business users can use to manage their tasks.

- The user portal (Figure 7-3) enables end users to claim tasks, alter task properties, and take actions on the tasks.
- The manager portal (Figure 7-4) enables managers to supervise the work done by the users whom they manage. It provides peer group views of tasks, their types, and statuses. It also allows the manager to view the details of subsets of tasks.

You can customize these portals or embed one or more of these portlets into your own portal.

Figure 7-3 Worklist User Portal

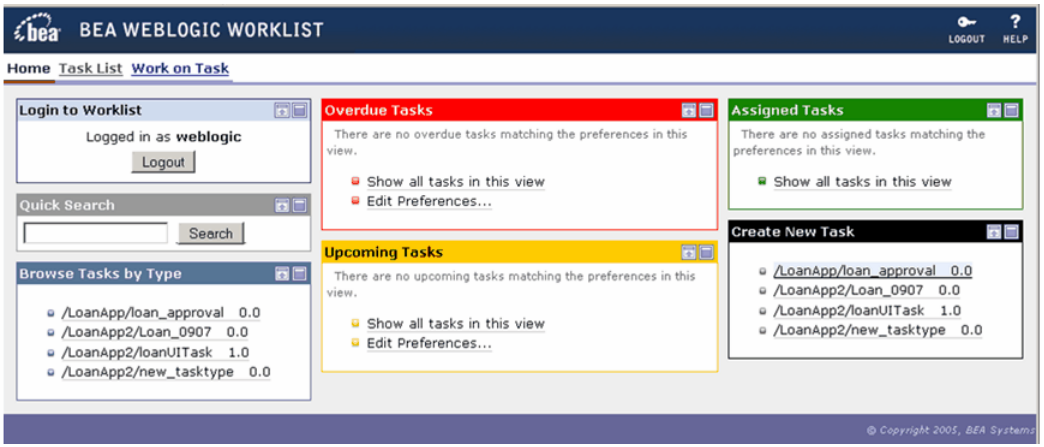
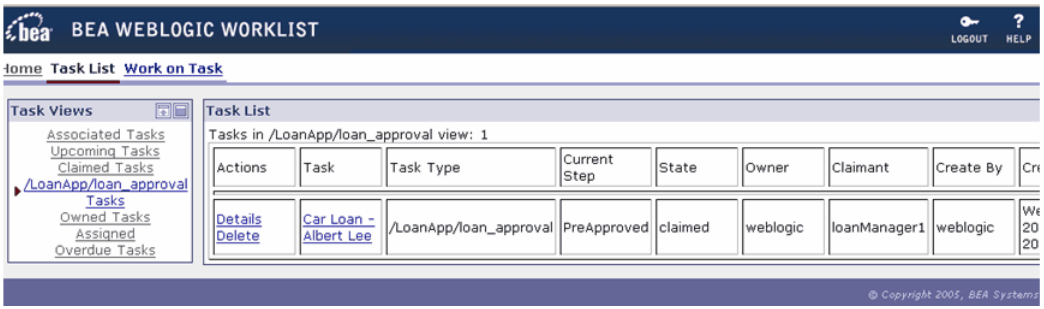


Figure 7-4 Worklist Manager Portal



For more information, see *Using Worklist*.

CHAPTER 8

Trading Partner Integration

WLI enables you to automate and manage relationships with your trading partners so that you can streamline your business processes with customers, suppliers, distributors, and other partners to get a top-down view of business transactions across the value chain.

Table 8-1 summarizes the trading partner integration capabilities of WLI.

Table 8-1 Trading Partner Integration in WLI

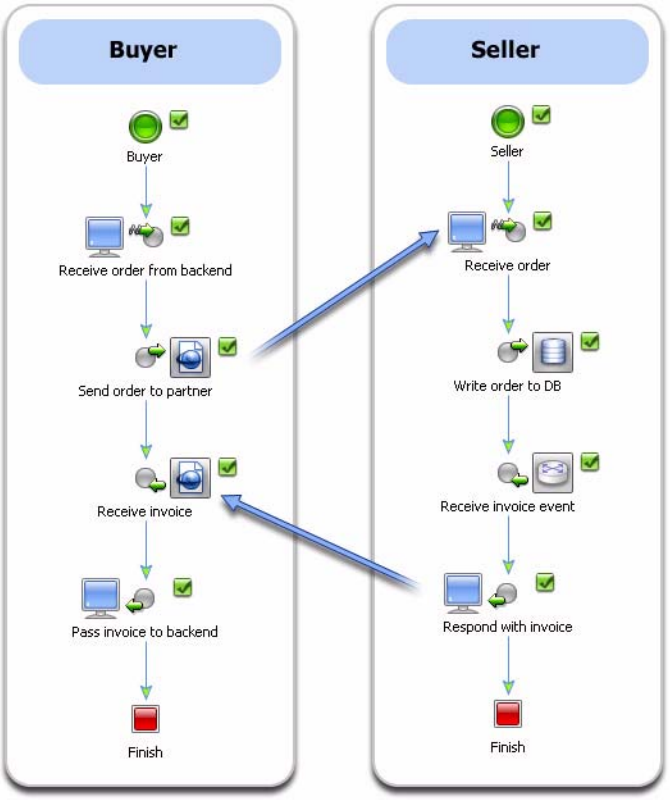
Feature	Description
Visual public and private process integration	WLI leverages the unified programming model and run-time framework of Workshop to provide end-to-end process integration with easily implemented controls and templates.
Support for leading B2B industry protocols and standards	WLI supports the following protocols and standards: • ebXML 1.0 and 2.0. • RosettaNet 1.1 and 2.0. • Web services.

Table 8-1 Trading Partner Integration in WLI (Continued)

Feature	Description
Trading Partner Management (TPM) and repository access	WLI provides sophisticated TPM capabilities through the WLI Administration Console. Administrators can easily manage a central repository of trading partner profile information, including protocol bindings used for secure message exchanges between trading partners, services representing public processes, security, and bulk import and export capabilities. Authorized processes and web services can dynamically access trading partner information by using easily implemented controls. In addition to the administration console, MBean APIs are provided so that third-party MBean clients can be written to access the TPM repository.
Easy access to run-time information	WLI provides flexible run-time tracking, audit, and reporting capabilities to show a top-down view of trading partner activities and business transactions across the value chain.
High performance and availability	WLI supports fast and reliable exchange of business messages between trading partners, and supports clustered configuration for scalability and failover, message persistence, low-level acknowledgments and receipts, and transactional integrity.
High security, auditing, and nonrepudiation	WLI ensures private, secure, and reliable exchange of business messages among trading partners by using transport-level security with SSL and message-level security with digital signature and encryption. The certificates and private keys are stored in protected keystores, and the passwords are stored in encrypted form in the WLI PasswordStore.
Interoperability	WLI operates in conjunction with a wide range of B2B servers from other vendors. For trading partners who want a zero-install solution, WLI can be extended easily to offer a browser or FTP interface.

Figure 8-1 shows an example of basic interactive business processes between trading partners.

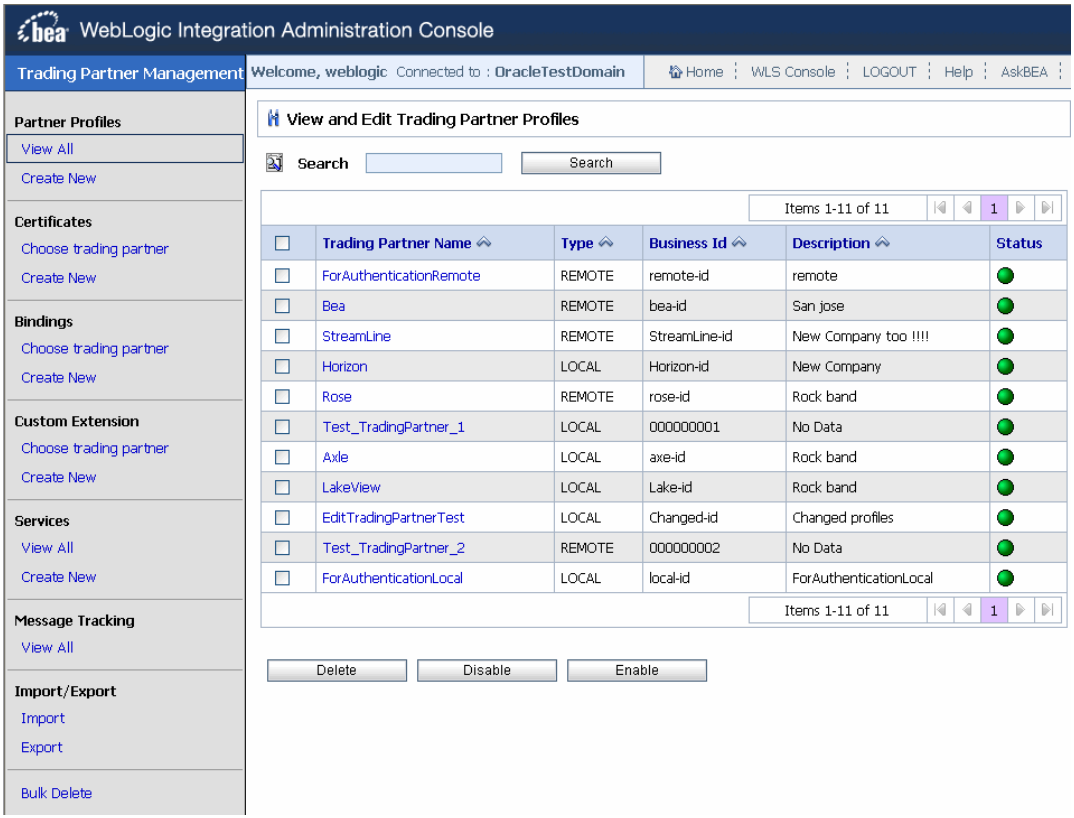
Figure 8-1 Interactive Business Processes Between Trading Partners



The buyer process sends an order to the seller by using a business protocol (ebXML or RosettaNet) that is agreed upon. The seller process receives the request, writes the order to a database, receives an invoice from an internal back-end system, and then sends the invoice to the buyer process by using the same business protocol. Such a message exchange is called a **conversation**. The initiating trading partner (buyer in this case) is the **initiator** and the responding trading partner (seller in this case) is the **participant**.

Figure 8-2 shows a partial view of the TPM home page in the WLI Administration Console, which allows administrators to manage trading partner profiles, security certificates, protocol bindings, services, message tracking and auditing, trading partner activity, and so on.

Figure 8-2 Trading Partner Management in the WLI Administration Console (Partial View)



For more information about trading partner integration, see *Introducing Trading Partner Integration*.

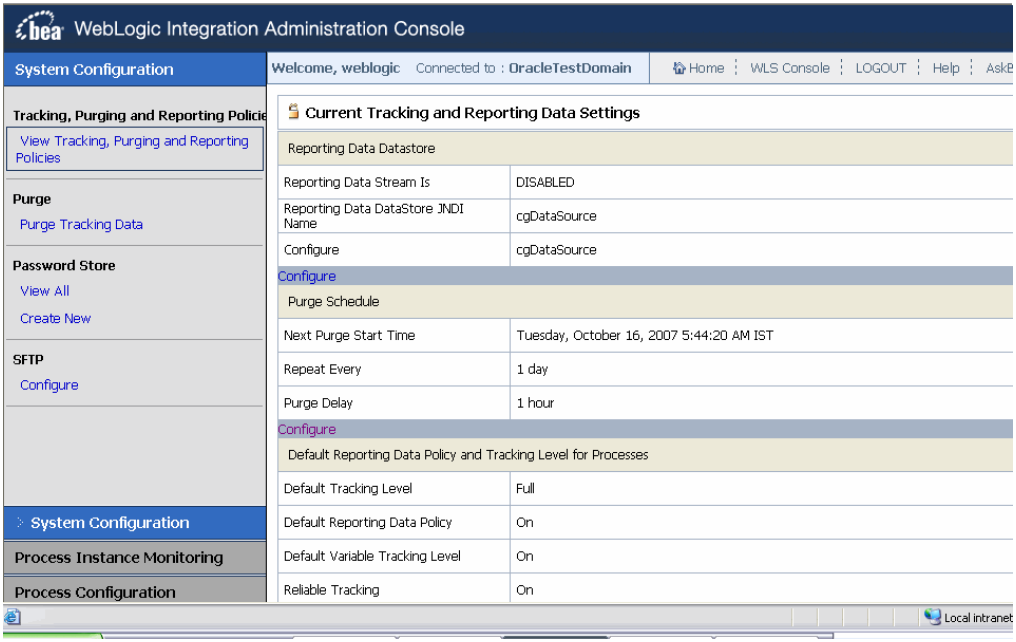
CHAPTER 9

Administration and Management

WLI provides a simple, secure, portlets-based administration console for run-time management and analytics. The console is easy to use and enables users to navigate quickly across modules of the console.

Note: This chapter provides an overview of the WLI administration console. For more information, see *Using the WebLogic Integration Administration Console*.

Figure 9-1 WLI Administration Console: Portlets-Based (Partial View)



The administration console allows centralized configuration, maintenance, and monitoring of integrated resources, including audit trails and the associated security and role information. It is extensible with JMX interfaces to third-party tools. The administration console is designed for application administrators, who need an integration-focused view of business processes and messaging activity in order to monitor deployed applications.

WLI separates run-time administration from offline analytics by maintaining two logical database stores.

- The online administration database contains run-time data about the integration engine, process states, and message history. This repository is designed for performance – to scale and retrieve information as quickly as possible while maintaining data in an optimized format. According to configurable archiving policies, the online repository is periodically archived to an offline data store. Data archiving enables analysis of process data by third-party tools through SQL queries on the archived databases.
- Task history is maintained in a separate offline store called the worklist reporting store. This store can be either the default reporting store provided with the worklist or a custom reporting store that you provide.

Table 9-1 lists the features of each module of the WLI administration console.

Table 9-1 Modules of the WLI Administration Console

Module	Purpose
Process Instance Monitoring	• View summary statistics about system health. • View summary and detailed status information for specific process instances. • View interactive and printable process instance graph. • Terminate or suspend instances, resume previously suspended instances, and unfreeze frozen instances. For more information, see Process Instance Monitoring in *Using the WebLogic Integration Administration Console*.
Process Configuration	• View process type information and locate specific processes for configuration. • View and update process type properties, such as the display name, tracking level, and reporting data policy. • View and update the security policies for a process. • Activate and deactivate a nonversioned process. • Configure the activation time for a newly deployed process version, or rollback to a previous version. • View an interactive or printable process type graph. • View and update the selectors used to dynamically set control attributes for a process control or service broker control. For more information, see Process Configuration in *Using the WebLogic Integration Administration Console*.
Message Broker	• View a list of channels, with the number of subscribers and processed messages for each. • View channel properties and set channel security policies. • View the subscribers to a channel and quickly access a list of the subscriber process instances. • View channel summary statistics (number of active channels, subscribed channels, and dead letter count). • Reset the message counter. For more information, see Message Broker in *Using the WebLogic Integration Administration Console*.

Table 9-1 Modules of the WLI Administration Console (Continued)

Module	Purpose
Trading Partner Management	• Manage and monitor trading partner profiles. • Manage services and service profiles that constitute the processes that are offered or called by trading partners. • Set message tracking criteria, and view summary and message content for tracked messages. • Import and export trading partner management data. • View summary statistics about the level of trading partner activity. For more information, see Trading Partner Management in *Using the WebLogic Integration Administration Console*.
System Configuration	• View and set the purge schedule. • Start and stop the purge process. • Enable and disable transmission of data to an offline data store. • View and set the JNDI name for the data store used to store data offline. • View and set the default tracking level and reporting data policy for processes. • Create, view, and change password aliases. For more information, see System Configuration in *Using the WebLogic Integration Administration Console*.
Event Generators	• Create and deploy new event generators. • Add channel rules to existing event generators. • Reset the read and error counters. • Suspend and resume deployed event generators. For more information, see Event Generators in *Using the WebLogic Integration Administration Console*.
XML Cache	• Add entries to the XML cache. • Change and delete XML cache entries. • View the code for XML cache entries. For more information, see XML Cache in *Using the WebLogic Integration Administration Console*.

BPEL Import and Export

Business Process Execution Language (BPEL) is a web service orchestration standard that can be used to define transactional business processes. With BPEL, activities that are part of a business process can be expressed as web services. These services can then be orchestrated to ensure control over the entire process. Processes written in BPEL are stored as `.bpel` files and can be executed on any platform or product that is compliant with the BPEL specification.

WLI provides BPEL Import and Export tools to enable design-time interoperability with other tools that support the BPEL 1.1 and 2.0 specifications.

In certain cases, however, run-time semantics are not guaranteed, especially when there are functional mismatches between the business process and BPEL, or between various expression languages, including differences between XQuery, XPath, and XSLT. Run-time semantics are also not guaranteed when they involve vendor extensions, external artifacts, or environment settings. For the above reasons, the imported and exported files must be reviewed and modified based on requirements to make sure that they execute correctly.

BPEL Import Tool

You can use the BPEL Import tool to import a BPEL 1.1- or 2.0-compliant file into a business process file, where it can be used in Workshop.

While the main orchestration logic of the BPEL file is imported into a business process file, it is not expected that the imported business process file will be immediately executable in Workshop. You will need to manipulate the business process file in the WLI Eclipse IDE, before it can be executed as a business process.

BPEL Export Tool

You can use the BPEL Export tool to export the semantics of a business process file into BPEL, where it can be used in a BPEL 1.1- or 2.0-compatible design environment.

While the main orchestration logic of the business process is exported to BPEL, it is not expected that the exported BPEL file will be immediately executable in the target environment. You will need to manipulate the BPEL file in the target environment to execute the exported process or to get close to the run-time semantics.

For more information, see *BPEL Import and Export User Guide*.

CHAPTER 11

Integration with Back-End Systems

BEA SmartConnect provides native connectivity to Enterprise Resource Planning (ERP) systems from AquaLogic Service Bus (ALSB).

WebLogic Integration (WLI) leverages SmartConnect 2.6 through AquaLogic Service Bus (ALSB) 2.6 RP1 by using standard protocols such as web services, JMS, and HTTP. You can use the control mechanism of WLI to connect to ALSB. In addition, you can leverage the tighter integration between WLI 10.2 and ALSB 3.0 to connect to ERP systems.

For more information about SmartConnect 2.6, see *BEA SmartConnect 2.6 Documentation*. For more information about integration between WLI 10.2 and ALSB 3.0, see Integration with AquaLogic Service Bus.

Note: In the past, WLI supported Application View controls to access J2EE Connection Architecture (JCA)-based adapters built on the Application Integration (AI) framework of WLI. The AI framework was deprecated in WLI 9.2 and is discontinued in WLI 10.2. JCA adapters are still supported, but customers using AI-based adapters are encouraged to start using SmartConnect 2.6 for connectivity with SAP and Siebel applications.

The following table summarizes the application integration features of WLI.

Table 11-1 Application Integration Features

Feature	Description
Application Integration	• BEA SmartConnect provides the native integration to ERP systems of SAP and Siebel. • Standards-based architecture for hosting J2EE CA-based adapters.
Adapters	• J2EE CA 1.0-based adapter infrastructure with extensions. • Service adapters expose application services to WLI. • Event adapters publish asynchronous, unsolicited messages from the application to the message broker.
XML Metadata Cache Management	• The XML metadata cache is a global, domain-wide cache. Data stored in this cache can be accessed by any business process that is deployed in the domain. You can use the cache to share data within a cluster. You can manage the cache through the WLI administration console or by using the MBean API, with which you can create your own NetUI-based consoles. • The XML metadata cache stores configuration data in the form of key-value pairs, where the key is of type String and the value contains XML data. Data from the cache is made available permanently through file-based storage. • A new XML metadata cache file is created for each XML document that is added to the cache. • WLI provides an XML metadata cache control, which you can use in business processes to retrieve data from the cache. The control uses the key (from the key-value pair) to retrieve associated XML metadata from the cache. For more information, see XML Cache in *Using the WebLogic Integration Administration Console*.

CHAPTER 12

Eclipse-Based IDE

WLI contains an IDE based on Eclipse 3.2.2. The following are some of the Eclipse-based elements that you can use in the WLI IDE:

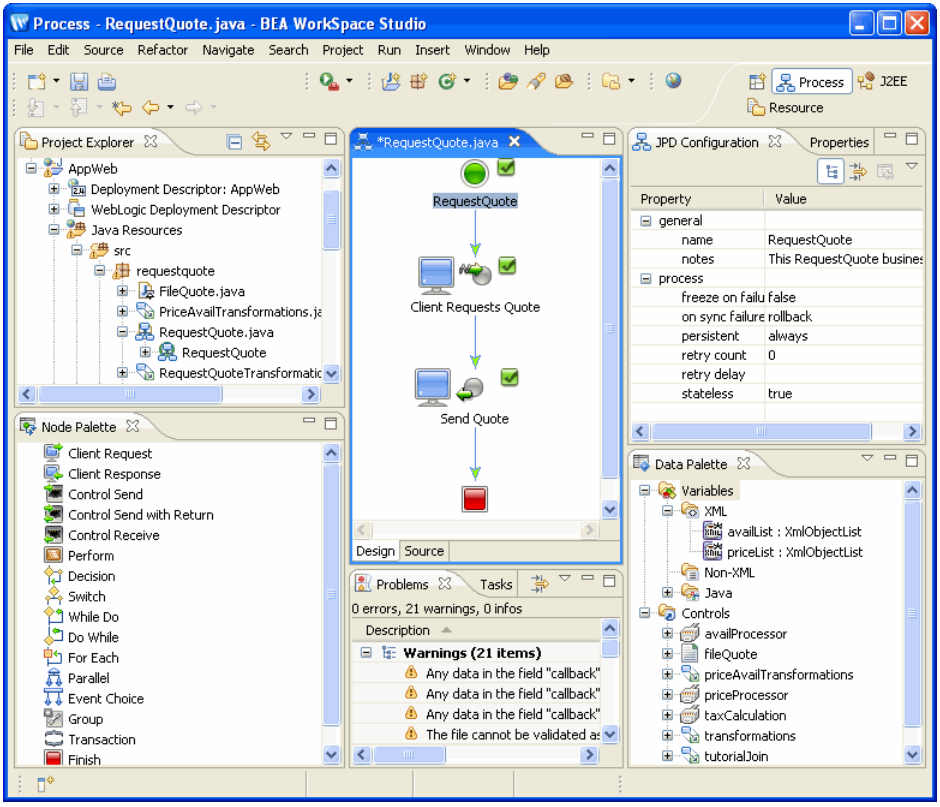
- Workbench
- Workspace
- Editors
- Views
- Resources
- Facets
- Perspectives

Workbench

The workbench is the development environment. It is the central place for creation, management, navigation, and control of various workspace resources. You can work on several projects simultaneously without reopening the IDE for each project.

Figure 12-1 shows the WLI workbench.

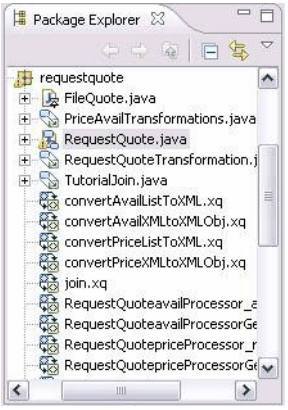
Figure 12-1 Workbench



Workspace

The workspace is the root directory that holds the projects, folders, sub-folders, files, libraries, and other artifacts. It also contains a `.metadata` folder that contains the Eclipse log files and plug-in folder for the Eclipse workspace, as shown in Figure 12-2.

Figure 12-2 Workspace



Editors

The editor allows you to open and edit a specific type of file. It also helps you visualize the file content as required.

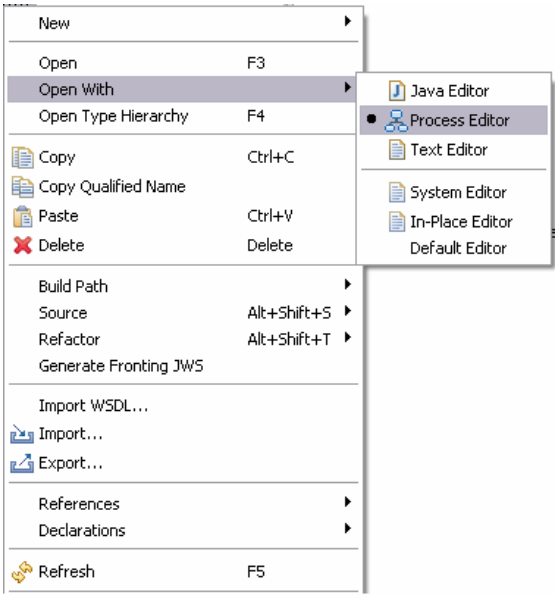
You can open multiple editors, but only one can be active at a time in the workbench. Editors are stacked by default, but you can tile and view them simultaneously. You can associate different files with different editors. When you double-click a file, the associated editor is invoked.

Some of the editors that WLI supports are:

- Process editor
- Task plan editor
- XQuery transformation editor
- Format builder editor

The editors are available in the central area of the workspace, as shown in Figure 12-3.

Figure 12-3 Editors



Views

Views support editors and provide an alternative presentation of workbench resources. You can use these views to easily navigate through the resources. Every view has its own menu, and, in most cases, contains toolbars as well. A view can appear alone or stacked with other views.

Some of the standard Eclipse views are:

- Package Explorer
- Navigator

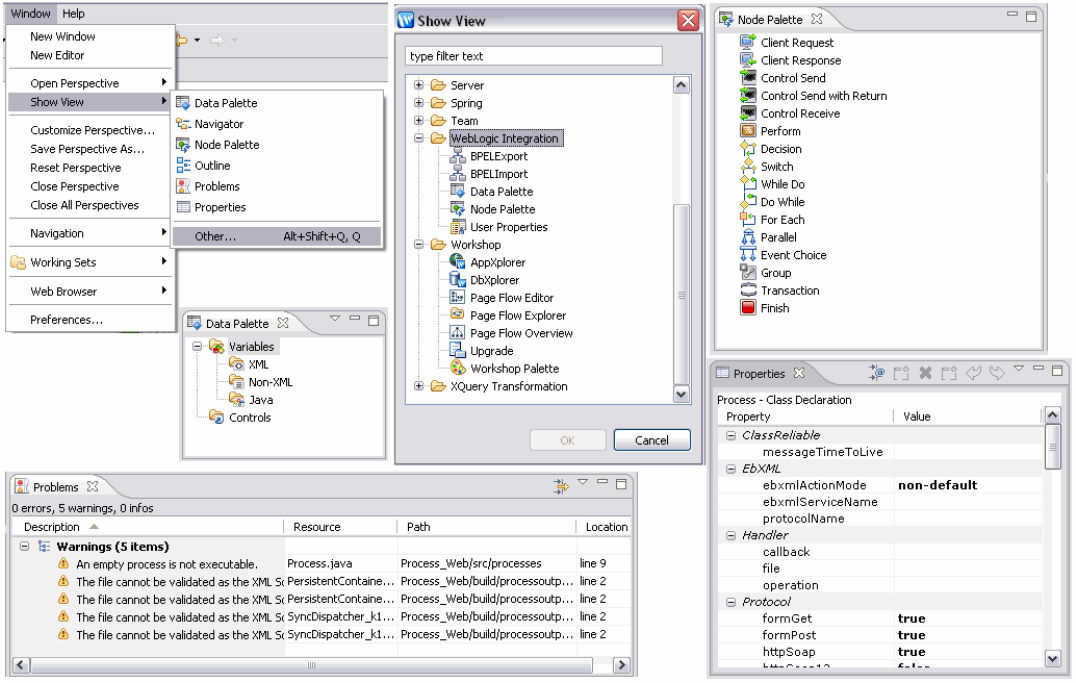
  You can use the Package Explorer and Navigator views to view projects and source files, as shown in Figure 12-5.
- Problems
- Properties
- Error log

Some of the WorkSpace Studio-specific views are:

- Node Palette
- Data Palette
- JPD Configuration
- Server

Figure 12-4 shows some of the views.

Figure 12-4 Views



Resources

The workspace contains three types of resources: projects, folders, and files.

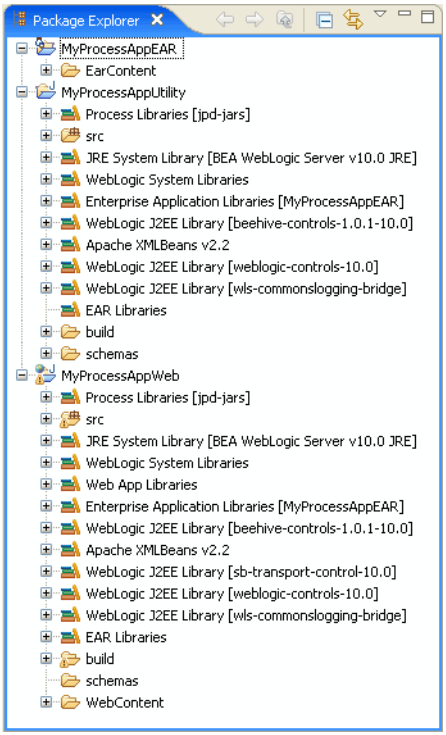
A project is a container within which you can organize the resources – folders and files – related to the project. Every project is associated with a metadata file called `.project`., which contains

information such as the project name, referenced projects, associated validators/builders, and applied facets.

Resources are organized in a hierarchical structure. The workspace contains projects, which, in turn, contain folders and files.

You can view the hierarchical structure of the resources within a project in the **Package Explorer** view, as shown in Figure 12-5.

Figure 12-5 Package Explorer View of Projects



Facets

When you create a project, you need to specify the type of the project in order to determine the capabilities of that project. For example, you need to add libraries, set compiler options, decide on publishing tasks, set the build path, add or remove validators, builders, and so on. Instead of making these settings manually every time you create a project, you can define a facet that

contains all of these settings. When you create a project, you can apply the required facet; the settings defined in the facet are applied automatically.

Every project has two core facets:

- An enabler facet, which specifies the type of project.
- An extension facet, which specifies the features added.

Facets are preconfigured into projects using the process application wizard. Figure 12-6 and Figure 12-7 show how you can add and remove facets to a project.

Figure 12-6 Select Facets

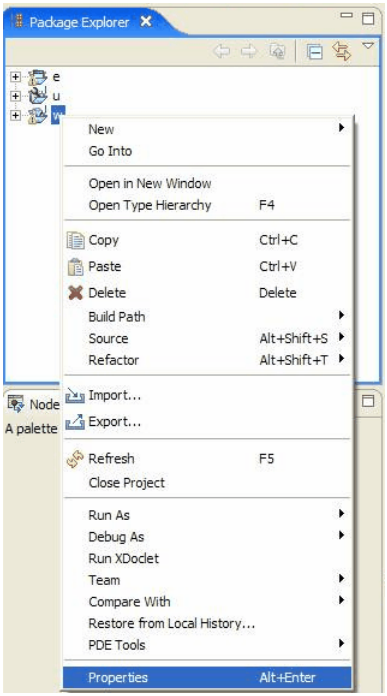
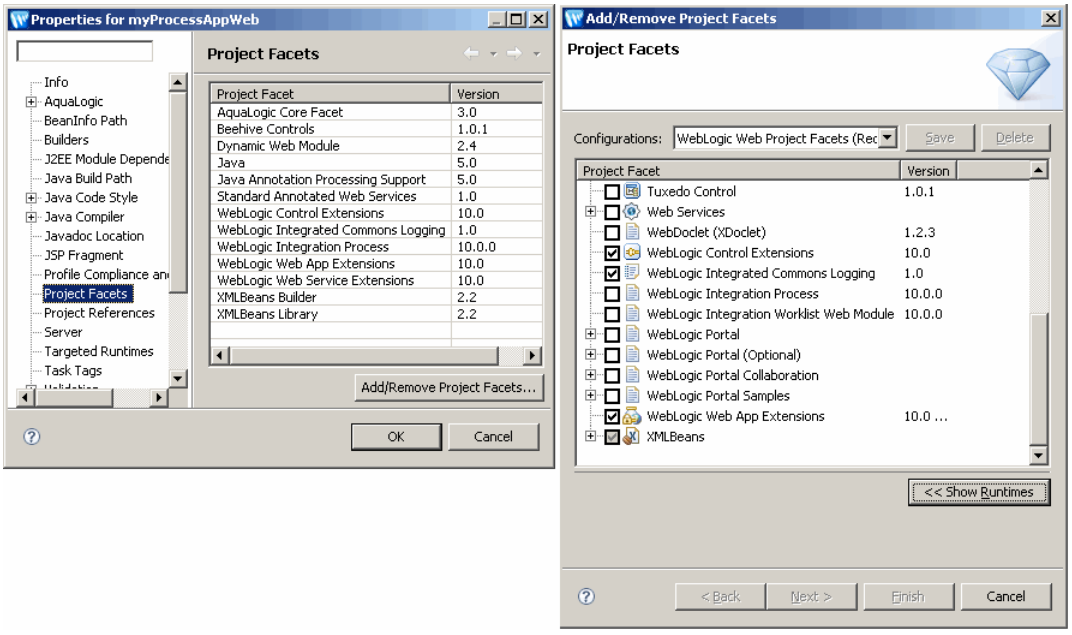


Figure 12-7 Add and Remove Facets



Perspectives

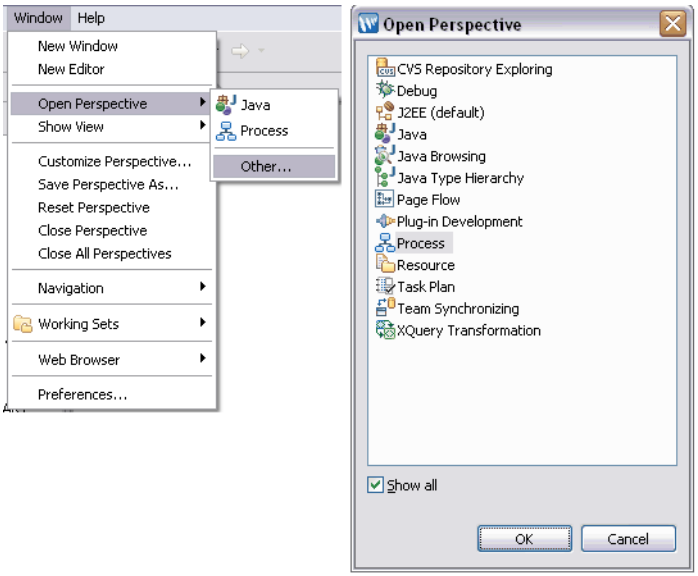
Perspectives define the initial set of views and their associated layout in the workbench. You can configure perspectives to define a collection of views, their layouts, and actions for specific tasks.

The WLI-specific perspectives are:

- Process
- Task Plan
- XQuery Transformation

Figure 12-8 shows how you can change the perspective.

Figure 12-8 Perspectives



For more information about Workshop for WebLogic, see the documentation at http://edocs.bea.com/wlw/docs102/index.html.

CHAPTER 13

Interoperability with AquaLogic Products

WebLogic Integration (WLI) interoperates with the following AquaLogic products:

- AquaLogic Service Bus (ALSB)
- AquaLogic Enterprise Repository (ALER)
- AquaLogic Enterprise Security (ALES)
- AquaLogic Data Services Platform (ALDSP)
- AquaLogic Business Process Management (ALBPM)

The interoperability between WLI and AquaLogic products provides a first step toward BEA Workspace 360 and BEA microService Architecture (mSA). In addition, integration with ALSAM and ALER provides capabilities for implementing SOA.

Integration with AquaLogic Service Bus

ALSB is part of the BEA AquaLogic family of service infrastructure products. It manages the routing and transformation of messages in an enterprise system. Combining these functions with its monitoring and administration capability, ALSB provides a unified software platform for implementing Service-Oriented Architecture (SOA). ALSB integrates seamlessly with other BEA products – WLS, WLP, and AquaLogic Data Services Platform (ALDSP) – for creating, consuming, and orchestrating services.

Integration of WLI with ALSB provides a cost-effective solution for building, connecting, and managing integrated process-driven services within and outside the enterprise, by combining the power and flexibility of WLI with the high-performance, stateless mediation of ALSB.

This integration provides the following features:

Table 13-1 Integration with ALSB: Features

Feature	Description
Ease of installation	• You can install WLI and ALSB in the same *BEA_Home*. • You can deploy WLI and ALSB applications in either a single domain (with a unified run time) or in separate domains. • This feature helps you reduce hardware cost, and leverage common resources, libraries, and plug-ins.
Unified design environment	• ALSB and WLI projects can be created in the same workspace. • Developers can navigate easily between ALSB and WLI design perspectives.
Tight integration with ALSB	• WLI processes can invoke external services through ALSB proxy services via ALSB Transport (ALSBT) controls. • External clients can invoke WLI processes as ALSB business services. • Developers can, at design time, search for RMI-based ALSB proxies and use them in WLI processes. • Security and transaction contexts are seamlessly propagated from ALSB to WLI and vice versa.
Aggregated deployment	• While deploying applications, you can deploy WLI and ALSB projects from the same workspace.

WLI and ALSB are also available as a cost effective bundle called AquaLogic Integrator. For more information, see the documentation at http://e-docs.bea.com/alint/docs30/alintuser/.

For more information about ALSB, see http://e-docs.bea.com/alsb/docs30/.

Integration with AquaLogic Enterprise Repository

BEA AquaLogic Enterprise Repository (ALER) is a SOA repository that provides the tools to manage and govern the metadata for any type of software asset, from processes and services to patterns, frameworks, applications, components, and data services. ALER maps the relationships

and interdependencies that connect those assets to improve impact analysis, promote and optimize their reuse, and measure their impact on the bottom line.

Note: ALER runs on WebLogic Server 9.2 MP1 only; so it cannot be installed in the same *BEA_HOME* as WLI. Once ALER is installed, however, you can use the functionality in the WLI IDE to connect to the ALER instance.

In WLI, you can do the following:

- Search for services stored in ALER and use them in WLI

  You can select services from ALER and use them in WLI processes. You can, for example, search for a specific web service, retrieve the WSDL from ALER, and use it to generate a service control in a WLI process. You can search for assets based on keywords or the type of service (WSDL- or XML-based).

  When a service that is stored in ALER is modified, ALER alerts existing users of the service about the change.

- Store metadata about WLI artifacts in ALER

  You can store metadata about WLI assets in ALER. The metadata includes information that can be used to make the WLI asset discoverable by other products for reuse.

For more information about ALER, see http://edocs.bea.com/aler/docs30/index.html.

Integration with AquaLogic Enterprise Security

BEA AquaLogic Enterprise Security (ALES) is a fine-grained entitlement management solution that combines centralized policy management with distributed policy decision-making and enforcement. This combination provides management and control of your critical applications and resources with uncompromised performance and reliability, allowing you to adapt to changing business requirements quickly and easily.

Integration of WLI with ALES allows administrators to implement policy-driven security, providing increased security for application- and system-level resources.

Administrators can leverage the features of ALES, such as the following:

- Manage users, groups, and roles, and configure policies for applications from a single ALES administration console, regardless of whether the applications are deployed on a single WLI domain/server or on multiple WLI domains/servers.
- Create user- and group-based policies for WLI in addition to role-based policies.

- Create policies with conditions. A user can, for example, be permitted to start a process only on a week day.
- Create policies with attributes. An employee can, for example, be permitted to start a process or execute a node of a process for only as long as the employee is in the finance department and has the role of a manager.
- Use ALES run-time APIs to implement security (authentication, authorization, and so on) in WLI applications.

For more information about ALES, see http://edocs.bea.com/ales/docs30/index.html.

Integration with AquaLogic Data Services Platform (ALDSP)

ALDSP provides the tools and frameworks necessary for rapid development and deployment of data services. Data services encapsulate the logic for reading, writing, and transforming information, insulating data consumers from having to contend with multiple data source formats and connection mechanisms.

WLI applications can use the ALDSP control to access data services that are deployed using ALDSP.

Note: The plug-in for the ALDSP control must be installed manually.

For more information about ALDSP, see http://edocs.bea.com/aldsp/docs30/index.html.

Integration with AquaLogic Business Process Management (ALBPM)

ALBPM integrates the modeling, implementation, execution, and monitoring of end-to-end business processes to support continuous optimization of the entire business process life cycle.

WLI processes (JPDs) can be exposed as web services, which can then be called by ALBPM applications.

For more information about ALBPM, see http://edocs.bea.com/albsi/docs60/index.html.

Standards Supported by WLI

The following table lists the standards that WebLogic Integration supports.

Table 14-1 Standards Supported by WLI

Standard	For more information, see ...
Standards supported by WebLogic Server 10.0	"Standards Support" section in WebLogic Server Release Notes
Standards supported by Workshop for WebLogic 10.2	"Support for Fundamental Frameworks and Tools" section in BEA Workshop User's Guide.
XML Schema 1.0	http://www.w3.org
XPath 2.0	http://www.w3.org
XQuery 1.0 W3C Working Draft 2002 and Recommendation 2004	http://www.w3.org
RosettaNet 1.1 and 2.0	http://www.rosettanet.org
ebXML 1.0 and 2.0	http://www.ebXML.org
WS-BPEL 1.1 and 2.0	http://www.oasis-open.org
SSH-FTP	http://www.ietf.org/