IBM WebSphere Application Server Enterprise V5 — Next-Generation Application Platform

Overview

WebSphere® Application Server Enterprise V5 delivers a powerful, next-generation application server. WebSphere Studio Application Developer Integration Edition for Linux and Windows® V5, separately available, complements this powerful application server with an integrated development environment. Together, these programs offer a platform designed to deliver on-demand e-business applications by simplifying build-to-integrate tasks, accelerating large-scale application development, and enabling real-time application flexibility.

This enterprise solution can integrate new and existing applications and resources, reduce IT development costs, and improve the use of development resources. It can also increase revenue and customer satisfaction, and improve time-to-market.

WebSphere Application Server Enterprise for Developers V5 provides a "unit test" environment for testing enterprise applications. With WebSphere Application Server Enterprise for Developers, programmers can test and verify the quality of their code to ensure that it meets certain criteria.

Key Prerequisites

The following are operating system requirements. Refer to Hardware Requirements for information on server requirements.

For more information on installation requirements, visit:


One of the following operating systems and, for viewing the product documentation, a Web browser that supports HTML 4 and Cascading Style Sheets (CSS):

- Windows NT® Server V4 with Service Pack 6a
- Windows 2000 Server or Advanced Server with Service Pack 3, or later
- Sun Solaris V8 at a maintenance level of July 2002, or later
- AIX® V4.3.3 with the 4330-10 recommended maintenance package or AIX 5.1 with the 5100-02 recommended maintenance package, or later
- Linux distribution for Intel® Red Hat Advanced Server 2.1, SuSE 7.3, or SLES 7 Linux based on kernel 2.4
- Linux for zSeries™ distribution SLES based on kernel 2.4

Planned Availability Dates

WebSphere Application Server Enterprise V5:

- February 21, 2003, electronic
- March 14, 2003, build to plan
- March 28, 2003, build to order

WebSphere Application Server Enterprise for Developers V5:

- February 21, 2003, electronic
- March 14, 2003, build to plan

At a Glance

WebSphere Application Server Enterprise V5, in conjunction with WebSphere Studio Application Developer Integration Edition for Linux and Windows, delivers a next-generation application server and development environment designed to:

- Simplify build-to-integrate tasks by reducing IT complexity, reusing existing resources, and automating business processes
- Accelerate large-scale application development by leveraging the latest innovations that build on today’s J2EE standards
- Enable real-time application flexibility by taking advantage of dynamic application support that allows you to build applications that can adapt on demand

For ordering, contact:
Your IBM representative, an IBM Business Partner, or the Americas Call Centers at 800-IBM-CALL

Reference: YE001

This announcement is provided for your information only. For additional information, contact your IBM representative, call 800-IBM-4YOU, or visit the IBM home page at: http://www.ibm.com.
As a result of rebranding, two programs names were changed. WebSphere Application Server Enterprise V5 is the follow-on program to WebSphere Application Server Enterprise Edition V4.1. WebSphere Application Server Enterprise for Developers V5 is the follow-on program to WebSphere Application Server Enterprise Edition Development Runtime V4.1.

**WebSphere Application Server Enterprise V5**

WebSphere Application Server Enterprise V5 delivers a powerful, next-generation application server. WebSphere Studio Application Developer Integration Edition for Linux and Windows V5, separately available, complements this powerful application server with an integrated development environment.

Together, these programs offer a platform designed to deliver on-demand e-business applications by helping simplify build-to-integrate tasks, accelerating large-scale application development, and enabling real-time application flexibility.

- **Simplify build-to-integrate tasks** — help reduce IT complexity, reuse existing resources, and automate business processes through a powerful but simplified build-to-integrate framework

- **Accelerate large-scale application development** — leverage the latest innovations that build on today's Java™ 2 Platform, Enterprise Edition (J2EE) standards to deploy a high-performance e-business infrastructure designed to cut costs, build customer loyalties, promote business agility, and gain a competitive advantage

- **Enable real-time application flexibility** — take advantage of dynamic application support that allows you to build applications that can adapt on demand, to the ever-changing world of e-business

WebSphere Application Server Enterprise for Developers V5 provides a "unit test" environment for testing enterprise applications.

**Simplify Build-to-Integrate Tasks**: Companies today face a growing problem as they begin to explore new e-business initiatives. The past 40 years of IT evolution have left them with an enterprise-computing infrastructure that is heterogeneous where business logic and application data are often scattered throughout the organization. Much of the business logic resides in databases, packaged applications such as Enterprise Resource Planning (ERP) systems, or in back-end systems such as CICS. Other business logic can be found in existing Java and J2EE applications and Web services. Furthermore, companies face constant pressure to create new applications in order to cut costs, build customer loyalties, and gain competitive advantage.

Companies need a way to reuse their existing software assets and to leverage the power of Web services in the development of new J2EE-based applications.

WebSphere Application Server Enterprise and WebSphere Studio Application Developer Integration Edition help companies to reduce IT complexity, reuse existing resources, and automate business processes through a powerful but simplified build-to-integrate framework.

- **Service-oriented architecture** — can reduce the complexity of large-scale application development and promote reuse by offering a standard way of representing and interacting with virtually all software assets

- **Integrated workflow** — increases development productivity and promotes reuse by enabling developers to visually choreograph interactions between software assets

- **Advanced transactional connectivity** — helps developers avoid custom coding by providing extended transactional support for the many challenges related to integrating existing software assets with a J2EE environment

**Service-Oriented Architecture**: Building new applications that integrate business logic and application data within the organization and with suppliers, partners, and customers is critical to the success of today's organizations. However, this integration remains complex, expensive, and risky.

A service-oriented architecture leverages open standards to represent virtually all software assets as services including legacy applications, packaged applications, J2EE components, and Web services. This approach provides developers with a standard way of representing and interacting with software assets. Furthermore, individual software assets become building blocks that can be reused in developing other applications.

Using this new service-oriented approach to integration, WebSphere Application Server Enterprise and WebSphere Studio Application Developer Integration Edition help reduce the complexity, cost, and risk of integration by providing a single, simple architectural framework based on Web services.

**Integrated J2EE-Based Workflow**: After a developer creates services out of an organization's software assets, the next logical step is to use those assets as part of a business process. Integrated J2EE workflow capabilities offer developers intuitive, flow-based development tools to take existing software assets and quickly define how those assets are used within a J2EE-based application.

For example, visual workflow tools can be used to combine inventory information from a packaged ERP application and J2EE components from a previously built customer-facing application with new business logic to create a new Web-based order entry application. The reach of the application can then be extended by exposing it as a Web service. The result can be faster development of new applications, improved consistency, and lower costs.

- **Visual process editor** offers intuitive drag-and-drop tools to easily compose and choreograph application interactions and dynamic workflows among J2EE components, Web services, existing applications, and human activities. Developers can quickly and easily build, debug, and deploy complex applications using powerful workflow tools and advanced messaging capabilities to streamline and automate business processes.

- **Human interaction** provides a ready-to-use Web client that allows workflow participants to access workflow applications and their business processes. This support includes activities that require a person to perform a task as a step in an automated business process. Specialized staff support allows the dynamic assignment of responsibilities based on existing organizational definitions.
Event triggering supports asynchronous events such as Web services or human interactions to be included as part of a business process.

Compensation pairs provide transaction “rollback-like” support for long-running, loosely coupled business processes that cannot be undone automatically by the application server. For example, the compensating transaction for an order that has already started manufacturing might be to put the completed item into inventory (rather than disassembling the item).

Flexible workflow design enables developers to design workflows using top-down, bottom-up, or meet-in-the-middle approach. Using top-down, developers can create skeleton processes that choreograph the sequence of events in a workflow without worrying about the underlying implementation. Building from the bottom up, developers first create the individual components and then use them as building blocks to defying a workflow. Meet-in-the-middle offers the flexibility of using both approaches at the same time.

Advanced Transactional Connectivity: Since its inception, the J2EE platform has made huge strides in providing enterprise level support for integration including support for messaging, security, and database access. The Java Connector Architecture (JCA) 1.0 standard begins to offer support for integrating with packaged and legacy applications. However, due to lack of adherence to data standards and limited transactional support, integrating with most back-end resources and legacy data can still be extremely difficult and risky.

WebSphere Application Server Enterprise and WebSphere Studio Application Developer Integration Edition offer advanced transactional capabilities.

Dynamic application adapter support builds and deploys rich, open standards-based application adapters for popular enterprise information systems such as SAP and IBM CICS.

Last participant support provides automated coordination for transactions that include two-phase commit resources and a single one-phase commit resource. This support helps to eliminate hand coding in this scenario and allows you to include one-phase commit resources.

Activity session services extend the scope of and group multiple local transactions. These local transactions can then be committed based on deployment criteria or through explicit program logic.

CORBA C++ Software Development Kit (SDK) integrates various C++ assets. This enables C++ clients to invoke J2EE components using CORBA technology and also lets WebSphere applications incorporate C++ assets behind CORBA wrappers.

Accelerated Large-Scale Application Development: Companies today strive to respond with flexibility and speed to customer demands, market opportunities, and external threats. However, for most companies, the time, cost, and complexity of large-scale application development make this goal extremely difficult to achieve.

WebSphere Application Server Enterprise and WebSphere Studio Application Developer Integration Edition leverage the latest innovations that build on today’s J2EE standards to help you deploy a high performance e-business infrastructure designed to cut costs, build customer loyalties, promote business agility, and gain competitive advantage.

Optimize Application Performance — Use powerful application profiling techniques, sophisticated deployment management, and advanced support for Web services to optimize performance and minimize downtime for applications that require highly available, high-volume, multiserver environments.

Enable next-generation development — Leverage the latest innovations that build on today’s J2EE standards to achieve greater control over application development, execution, and performance.

Increase development productivity — Take advantage of supported, prebuilt J2EE-based solutions to many of today’s biggest programming challenges.

Optimize Application Performance: Increasingly, organizations are using Web applications both internally and externally to incorporate customers, partners, and suppliers into their business processes. For these mission-critical processes, application performance can make the difference between competitive advantage and failure to compete. WebSphere Application Server Enterprise and WebSphere Studio Application Developer Integration Edition help you optimize performance and minimize down time for applications.

Application profiling delivers powerful new capabilities that allow you to carefully optimize the performance of applications without any impact on source code. This capability offers a mechanism for specifying the access intent of persistent entity Enterprise JavaBeans (EJBs), allowing them to interact differently with the run-time infrastructure, such as database, depending on the access intent (for example, read versus update) of the application that calls it.

Deployment manager addresses the needs of highly available, high-volume, multiserver environments through enhanced workload management and dynamic caching, centralized security, and performance management tools that distribute workload across multiple servers through sophisticated load balancing and clustering. The deployment manager also enables isolation of application servers to avoid single points of failure and provides first failure data capture to report and analyze problems as they occur.

Advanced Web services support includes a Universal Description, Discovery and Integration (UDDI) Registry that acts as a repository for storing business units that describe basic Web services. A Web Services Gateway enables Web services invocation by users from outside the firewall with the benefit of robust security protection. Advanced Web services support also extends the Web Services Gateway with a programming model that allows you to use the gateway in large-scale Web services implementations to serve as a bidirectional control point for critical tasks such as validation, logging, transformation, auditing, and metering.
Cheat sheets make new or complex tasks easy by providing a checklist for common development patterns. The cheat sheet invokes each step in the checklist and provides detailed outline help for each step as you need it.

Asynchronous beans offer exceptional performance enhancements for resource-intensive tasks by allowing a single request to be executed as multiple tasks or threads processed in parallel within the J2EE environment. Asynchronous scheduling facilities can also be used to process parallel processing request in “batch mode” at a designated time.

Object pools help increase application performance by allowing instances of objects to be reused, reducing the overhead associated with the instantiating and garbage-collecting the objects. Creating an object pool allows an application to obtain an instance of a Java object and return the instance to the pool when it has finished using it.

Increase Development Productivity: The time required to roll out new applications is a key concern across all industries. One way to help improve developer productivity is to reduce the need for handcrafted solutions that can be time-consuming, costly, and difficult to maintain. WebSphere Application Server Enterprise and WebSphere Studio Application Developer Integration Edition can improve developer productivity by leveraging the latest innovations that build on today’s J2EE standards to provide supported, prebuilt solutions to many of these challenges.

Extended messaging allows you to quickly create applications that integrate with other systems through a messaging infrastructure. This capability offers automated support for outbound (as well as inbound) messaging, allowing you to focus on business logic instead of complex messaging APIs. Handcrafting Java Message Service (JMS) code is no longer required. WebSphere Application Server Enterprise includes WebSphere MQ and WebSphere MQ Event Broker (under limited terms and conditions) to further extend your Java messaging infrastructure to take advantage of both programs’ qualities of service and to enable seamless integration with existing MQ infrastructures.

暂缓sheet make new or complex tasks easy by providing a checklist for common development patterns. The cheat sheet invokes each step in the checklist and provides detailed outline help for each step as you need it.

Enable Next-Generation Development: Ironically, J2EE’s main advantage, its specification, can also be its biggest disadvantage for developers building applications that require them to have more control over their applications than the J2EE specification provides. For those developers, WebSphere Studio Application Developer Integration Edition and WebSphere Application Server Enterprise enable next-generation development by leveraging the latest innovations that build on today’s J2EE standards.

- Asynchronous beans offer exceptional performance enhancements for resource-intensive tasks by enabling a single request to be executed as multiple tasks or threads processed in parallel within the J2EE environment. Asynchronous scheduling facilities can also be used to process parallel processing request in “batch mode” at a designated time.
- Startup beans allow business logic to be automatically executed when an application starts or stops. For example, they might be used to prefill application-specific caches, initialize application-level connection pools, or perform other application-specific initialization and termination procedures.
- Scheduler service helps minimize IT costs and increase application speed and responsiveness by maximizing utilization of existing computing resources. The scheduler service processes workloads using parallel processing, sets specific transactions as high priority, and schedules less time-sensitive tasks to process during low traffic off-hours.
- Object pools help increase application performance by allowing instances of objects to be reused, reducing the overhead associated with the instantiating and garbage-collecting the objects. Creating an object pool allows an application to obtain an instance of a Java object and return the instance to the pool when it has finished using it.

Enable Real-time Application Flexibility: Maintaining competitive advantage in today’s changing business environment requires companies to respond quickly to customer demands, market opportunities, and external threats. Very often this means making frequent updates to e-business applications to reflect changes in market conditions or to provide access to e-business information. Unfortunately, these updates usually take a great deal of time: time to bring down the application, time to make programming changes, time to test the new application, and time to redeploy.

To enable real-time application flexibility, WebSphere Application Server Enterprise and WebSphere Studio Application Developer Integration Edition offer dynamic application support to enable you to build applications that can easily adapt to the ever-changing world of e-business on demand™.

- Business rule beans offer a powerful real-time framework for defining, executing, and managing business rules that encapsulate business policies that vary based on changes in the business environment. For example, a simple business rule might be, “If a customer’s shopping cart is greater than $X, then offer a Y% discount.” Once the business rule is defined, a developer or a business analyst can update the business rule at run time using a straightforward interface without the need to bring the application or server down.
- Dynamic query service delivers new application flexibility by allowing you to dynamically build and submit queries that select, sort, join, and perform calculations on application data at run time. Dynamic query service provides the ability to pass in and process EJB Query Language queries at run time, eliminating the need, as with today’s EJB 2.0 standards, to hard-code required queries into the deployment descriptors during development.

For more information on WebSphere Application Server Enterprise and WebSphere Studio Application Developer Integration Edition, visit:

http://www.ibm.com/software/webservers/appserv/enterprisedevelopment
http://www.ibm.com/software/ad/studiointegration
Statement of Direction

Adapter Support

Planned for availability in the first half of 2003, WebSphere Application Server Enterprise V5 in conjunction with WebSphere Studio Application Developer Integration Edition V5 intends to provide support for a suite of more than 30 WebSphere Business Integration Adapters, including adapters for SAP R3, PeopleSoft, Siebel, Oracle Applications, SWIFT, and other applications and technologies. WebSphere Business Integration Adapters should help customers achieve business agility by integrating applications and automating business processes. The IBM portfolio of adapters includes solutions for packaged applications, e-business architecture, legacy products, technology protocols, industry solutions, and trading partner connectivity.

IBM is driving open standards in application integration and connectivity, and is committed to introducing WebSphere Business Integration Adapters developed on the latest open standards, including J2EE Connector Architecture.

Business Process Execution Language for Web Services


BPEL4WS is the new emerging standard for the modeling and execution of business processes in an e-business environment. This standard is jointly supported by Microsoft Corporation, BEA Systems, Inc., and IBM. Refer to the press release, dated August 9, 2002, at:

http://www-916.ibm.com/press/prnews.nsf/jan/AD23C3FF0C20CD1585256C10004F7E52

Additional Support

IBM plans to make WebSphere Application Server Enterprise V5 and WebSphere Application Server Enterprise for Developers V5 available on HP-UX, Red Hat Linux on zSeries, and Linux Power PC at a later date. IBM plans to support native JMS on WebSphere Application Server Enterprise for zSeries, V5, at a later date. IBM also plans to make interoperability support for third-party ORB (including documentation and samples), additional Web services features, Application Assembly Workbench, and Performance Advisor available at a later date.

All statements regarding IBM’s plans, directions, and intent are subject to change or withdrawal without notice.
The use of business processes changes the way applications are built. The process engine allows an application architecture that separates the description of the business logic (the flow logic) from the implementation of the business functions. The resulting application structure is known as a business process-based application. The control flow is managed by a workflow management system, which is responsible for the invocation of the individual business functions according to the business logic.

Programming a business process-based application consists of two independent steps:

- Programming the components. The individual business functions are implemented as usual; that is, they are coded, bought, or already exist. The actual representation of a business function can be, for example:
  - An EJB method
  - A JCA connector call that provides connectivity to a back-end system, such as CICS
  - An operation of a WSDL port type, for example, a SOAP-based Web service

- Programming the business process. The business logic is described as a process that consists of the steps in the process that need to be performed, their relationship to one another, and their ordering constraints.

A business function is assigned to each of the steps in the process. A process step can be an elementary step or a composite one, resulting in a recursive programming model.

Business process-based applications offer advantages over traditional applications because they have a number of properties that are controlled by the middleware responsible for their execution. These properties are:

- Concurrency. If a process contains parallel branches, the middleware helps guarantee that the branches are executed concurrently in parallel threads, possibly even on different nodes in a cluster.

- Recoverability. If the system crashes while executing a process-base application, the execution of the application can be continued where it was left off — steps that have already been performed are not redone.

- Heterogeneous, distributed execution. The execution of the individual functions of a business process (its steps) can be distributed in a network, on heterogeneous operating systems, and on hardware platforms.

- Range of quality of service. The process engine supports non-interruptible processes (microflows), interruptible processes (macroflows), and combinations of the two.

---

Applications: Traditionally, an application contains code for the implementation of individual business functions (such as “create order entry”) and code for the flow logic of the application according to the business requirements (such as “high-value order messages have to be separately approved”). Whenever the business requirements change, the code must be changed. Usually these changes are to the business flow logic and only rarely do they affect the implementation of the individual business functions.
Involving People: Many enterprises are looking for ways to automate their business processes. Fully automated processes, without human interaction, can typically be executed at low cost, with high throughput, and short response times.

However, many real-world processes cannot be fully automated. For example, special approvals might have to be given by an authorized manager, rather than automatically by a machine. Therefore, it is common for business processes to consist of a combination of automated steps and steps with human interaction.

As long as the automated steps are short and do not require human intervention, this type of process can be done with high throughput and short response times. Although human interaction is part of the definition of the process, this does not impact its execution if those steps are not actually performed. The business-process paradigm easily supports the description of the entire business process, including the manual steps, as a single process. The process engine is responsible for the efficient execution of automatic processes and for the inclusion of human interaction to handle exceptions.

Process Choreographer provides a ready-to-use Web client that allows workflow participants to access workflow applications and their business processes.

Undoing Complex Operations Using Compensation: Today’s applications typically require transactional properties, such as a complex request is executed either in its entirety or not at all. For traditional transactions, this is described by the ACID properties: atomicity, consistency, isolation, and durability. It is achieved by a transaction manager, resource managers, and back-end systems working together according to the Extension Architecture (XA) standard or similar protocol. This cooperation ensures that the operations performed on behalf of a transaction are either all committed or all rolled back.

Often, however, a complex request cannot be run as an ACID transaction for a number of reasons:

- Back-end systems or resource managers might not be able to participate in the XA protocol. Updates to these systems are performed immediately and do not participate in the overall transaction. If the transaction fails, these updates are not undone; the processing of the complex request results in an inconsistent state.
- As long as a transaction has not been committed, none of the changes done on its behalf are visible to the outside world. The isolation property establishes that they only become visible when the transaction’s final state has been reached. This works well for short-running processes that invoke synchronous operations. However, when a process involves asynchronous steps (such as steps implemented by a back-end system driven by JMS messages or steps involving human interaction), intermediate results of the process must be made visible. This means that the JMS message must be sent out, the information about the step the person has to work on (the work item) must be made available, and so on. Therefore, the intermediate state of the process must be committed.
- There are actions in a business process that are inherently nontransactional. Sending a letter to a customer is such an operation: as soon as the letter has left the sender, there is no way to undo the operation.

In all of these examples, intermediate results of the business process are made available and they cannot be undone by simply rolling back an ACID transaction with the help of the transaction manager. Instead, another operation must be done that explicitly reverses the original operation:

- If a back-end system has been called to update data (for example, to increment a value), the system must be called again to reverse the process.
- If a letter has been sent in error to a customer, another letter must be sent to apologize for the error.

These operations might need to be cascaded if other operations have already started that use the results that have been wrongly made available.

To help you define undo operations and their automatic execution, you can specify the steps of a process as compensation pairs. This means that in addition to the standard forward operation, a backward operation is assigned to this activity.

The process runs using the forward operations. In addition, each invocation of a forward operation is logged with its input and output data in a compensation list. At any time during the execution of a process, you know which operations have been invoked and with which data.

If the process fails during its execution and has to be compensated, the compensation list is used to drive the backward execution of the process to reestablish the previous process state. For those operations that were executed during the forward execution of the process, the associated undo operation is invoked, and the original data is passed. For example, the undo operation might send a compensation letter to tell the customer that a previous letter was in error and should be ignored.

Business-to-Business (B2B) Processes: Business processes are typically not limited to applications and people within an enterprise, they also involve business partners. Interactions with these partners can be modeled easily as part of the process.

There are two types of interactions with business partners:

- Inbound requests. A request is sent by a business partner, either to initiate a business process or to pass data to a running business process as part of a conversation.
- Outbound requests. A request is sent to a business partner, either to initiate a business process there, or to send data as part of a conversation. A special case of an outbound request is the completion of a business process and the delivery of its results to the business partner.

For both types of requests, the interaction with business partners involves the use of a WebSphere component, WebSphere Services Gateway. The gateway is responsible for routing inbound Web-services requests to the appropriate WebSphere component such as the process engine, and routing outbound Web-services requests to the right business partner.

Business Rule Beans: Business rule beans are used to create and modify rules that keep pace with complex business practices. This enables your application’s core behavior and user interface objects to remain intact and untouched, even as business practices change.
The business rules beans (BRBeans) framework enables you to organize rules folders. Folders provide a structure similar to the file system on your computer’s hard drive; for example, rules can be placed in folders based on any criteria you want. A rule folder can contain any number of rules and other rules folders. In the BRBeans framework, each business rule is represented by an entity bean that persistently stores information related to that rule. Each business rule is assigned an appropriate rule name and stored in an appropriate rule folder.

A fully qualified name consists of the full path of the folder followed by a slash (/) and the name of the rule.

This fully qualified rule name is used by a trigger point to identify the rule to trigger. Trigger points are small pieces of code that interface with the BRBeans trigger-point framework to run business rules during application execution.

By default, trigger points can only trigger rules that are currently in effect based on the current date and time when the trigger point is called. A business rule has a start date and an end date that together define the interval during which the rule is in effect. This behavior can be overridden by specifying a date on a trigger point. This date is referred to as the “as of date.” If no start date is specified, the rule never expires. Dates and times with a precision of one second can be assigned using the rule management application.

When there is more than one rule with the same fully qualified name, all of the rules with that name that are currently in effect are triggered, and the results are combined using the combination strategy specified on the trigger point.

Asynchronous Beans, Object Pools, Startup Beans, and Scheduler Service: An asynchronous bean is a Java object or enterprise bean that can be executed asynchronously by a J2EE application, using the J2EE context of the bean’s creator. Asynchronous beans can improve performance by enabling a J2EE program to decompose operations into parallel tasks. Asynchronous beans enable the construction of stateful, “active” J2EE applications. These applications address a segment of the application space that J2EE has not previously addressed (that is, advanced applications that require application threading, active agents within a server application, or distributed monitoring capabilities).

An object pool enables an application to avoid creating new Java objects repeatedly. Most objects can be created once, used, and then reused at a later point. An object pool allows an object to be pooled while waiting for the point where it can be reused. These object pools are not meant to be used for pooling Java Database Connections (JDBC) or JMS connections and sessions. WebSphere Application Server Enterprise provides specialized mechanisms for dealing with those types of objects. These object pools are intended for pooling application-defined objects or basic SDK types.

A startup bean is a stateful session bean that can be loaded when an application starts. Startup beans enable J2EE applications to execute business logic automatically, whenever an application starts or stops normally. Startup beans are especially useful when used in combination with asynchronous bean features. For example, a startup bean might create an alarm object that uses JMS to periodically publish heartbeat messages on a well-known topic. This enables clients or other server applications to determine whether the application is available. The scheduler service enables tasks to be executed at a requested time. The following tasks can be scheduled:

- Invoke a session bean method
- Send a JMS message on a queue or topic

The scheduler service performs the task, repeating as necessary, according to the task’s metadata.

Dynamic Query Service: Consider using the dynamic query service when any of the following are true:

- You do not know the query search criteria until application run time.
- You need to return multiple query search criteria from a query (deployment queries allow only a single element to be specified in the SELECT clause).
- You want to perform aggregation in the query (deployment queries do not allow the use of aggregation function SUM, AVG, COUNT, MAX, or MIN in the top level SELECT of a query).
- You want to use value object methods or bean methods in a query statement.
- You want to interactively test an EJB query during development but do not want to repeatedly deploy your application each time you update a finder or select query.

If you have a query that has a high frequency of execution, you should define it as a finder or select method and consider using SQLJ as a deployment option for best performance. The dynamic query service always uses JDBC and must parse and process the EJB query at run time.

If you need security control over which queries a user can execute, you need to define the queries as finder or select methods and use EJB method authorization. The dynamic query service does not have fine-grained security control at this time. You can control who is permitted access to the remote query bean and local query bean, but once authorized, a user can execute any valid query and return any data in the server.

The dynamic query API is a stateless session bean. Using the dynamic query API is similar to using any other J2EE EJB application bean.

Application Profiling: Application profiling enables developers to identify particular units of work to the WebSphere V5 run-time environment, such that the run time can tailor its support to that unit of work’s requirements. Access intent is currently the only component of the run time to make use of application profiling. Application profiling, combined with access intent, enables developers to configure applications so that an entity bean can be loaded in one transaction with strong update locks, loaded in another transaction with no locks at all, and with significant portions of the object graph read ahead in a single database operation.

Application profiling introduces two new concepts in order to achieve this function:

- A task is a named unit of work within a distributed application. Unit of work, in this case, means a unique path within the application that may or may not correspond to a transaction or activity session. The name of the path is typically assigned declaratively to a J2EE client or servlet, or to the method of an enterprise bean. This point of configuration marks the head of a graph or subgraph identified by the name of the task; the task name flows from the head of the graph downstream on all subsequent IIOP requests, identifying each subsequent invocation along the graph as belonging to the developer-configured task.
A profile is a set of policies that are configured not only on the components of an application, but on a set of tasks as well. When an invocation on a bean (whether by a finder method, a cmr getter, or a dynamic query) requires data to be retrieved from the back-end system, the current task associated with the request is used to determine the exact requirement of the transaction: the same bean loads and behaves differently in the context of the task to profile mapping. Each profile provides the developer an opportunity to reconfigure the applications access intent. If a request is operating in the absence of a task, the run-time environment uses the access intent configuration external to the application profiles.

**Extended Messaging:** Extended messaging adds to the base JMS support by providing EJB 2.0 message driven beans, and the EJB component model, to allow the use of existing container-managed persistence and transactional behavior.

In addition to providing container-managed messaging (CMM), extended messaging provides new types of enterprise beans and administrative objects for messaging and new functions such as data mapping and late response handling. (CMM is sometimes used to represent all of extended messaging.)

Extended messaging uses the bean-managed messaging implementation to provide the JMS interfaces, which ensure that both bean-managed messaging and extended messaging use consistent JMS support.

An application that uses extended messaging can receive messages by using a receiver bean, either by the onMessage() method of a message-driven bean or by a stateless session bean that polls for a message from a named destination. With extended messaging and a message-driven bean, code within the bean can use the message to invoke business logic, either as a method within the same bean or another enterprise bean. Both the incoming message and the invocation of the receiver bean can be included within the scope of a transaction. For outbound messages, an application calls a sender bean that turns a method call into a JMS message that is sent asynchronously. These message beans are implemented as enterprise beans by WebSphere Application Server. Application developers can create these message beans using WebSphere Studio Application Developer Integration Edition.

With extended messaging, the JMS usage is simplified because JMS support is managed by the extended messaging service. This helps to effectively separate business logic from the messaging infrastructure. Also, the use of data mapping enables messages to drive existing or new enterprise beans that support extended messaging to be developed easily and hide the messaging infrastructure from developers.

**Internationalization Service:** In a distributed client/server environment, application processes can run on different machines, configured to different locales, corresponding to different cultural conventions; they can also be located across geographic boundaries. The J2EE platform provides support for application components executing on computers with different endian architectures and code sets, but does not provide dedicated support for application components that run on computers having different locales and time zones.

The conventional method for solving locale and time zone mismatches across remote application components is to pass one or more extra parameters on all business methods needed to convey the client-side locale or time zone to the server. Although simple, this technique has the following limitations when used in EJB applications:

- It is intrusive because it requires that one or more parameters be added to all bean methods in a call chain to locale-sensitive or time zone-sensitive methods.
- It is inherently error-prone.
- It is impracticable within applications that do not allow for modification, such as legacy applications.

The internationalization service addresses the challenges posed by locale and time zone mismatch without incurring the limitations of conventional techniques. It does this by systematically managing the distribution of internationalization contexts across the various components of EJB applications, including client applications, enterprise beans, and servlets.

The service works by associating an internationalization context with every thread of execution within an application. When a client-side component invokes a business method, the internationalization service interposes by obtaining the internationalization context associated with the current thread of the client-side process and attaching that context to the outgoing request. On the server side, the internationalization service again interposes by detaching the context from the incoming request and associating it with the thread of the server-side process on which the business method will execute, effectively scoped the context to the business method. The service propagates internationalization context on subsequent business method invocations in the same manner and thus distributes the context of the originating request over the entire chain of business method invocations.

This basic operation of scoping and propagation is defined precisely by internationalization context management policies. Every application component has a default policy, which can be overridden and tailored for servlets and enterprise beans at development time using the WebSphere application assembly tool. Internationalization policies specify whether an application component or its hosting J2EE container are to manage internationalization context. For container-managed components, the policy indicates which internationalization context the container will scope to invocations of that component. Server components configured to manage internationalization context, as well as EJB clients, must use the internationalization context API to manage the internationalization context elements scoped to their invocations.

At execution time, application components can use the internationalization context API to get any element of the internationalization contexts scoped to an invocation. To programatically access context elements, application components first resolve an internationalization context API reference, then invoke the appropriate API method to access the various context elements, such as the caller locale or the invocation time zone. These elements can be used in calls to Java 2 SDK internationalization API methods; for example, to perform localizations such as formatting messages, configuring dates, or comparing strings.

**WorkArea Service:** One of the foundations of distributed computing is the ability to pass information, typically in the form of arguments to remote methods, from one process to another. When application-level software is written over middleware services, many of the services rely on information beyond that passed in the application’s remote calls. Such services often make use
of the implicit propagation of private information in addition to the arguments passed in remote requests; two typical users of such a feature are security and transaction services. Security certificates or transaction contexts are passed without the knowledge or intervention of the user or application developer. The implicit propagation of such information means that application developers do not have to manually pass the information in method invocations, which helps make development less error-prone and the services requiring the information do not have to expose it to application developers. Information such as security credentials can remain secret.

The WorkArea service gives application developers a similar facility. Applications can create a work area, insert information into it, and make remote invocations. The work area is propagated with each remote method invocation, eliminating the need to explicitly include an appropriate argument in the definition of every method. The methods on the server side can use or ignore the information in the work area as appropriate. If methods in a server receive a work area from a client and subsequently invoke other remote methods, the work area is propagated with the remote requests. When the creating application is done with the work area, the application terminates the work area.

There are two prime considerations in deciding whether to pass information explicitly as a argument or implicitly by using a work area.

- Pervasiveness — Is the information used in majority of the methods in an application?
- Size — Is it reasonable to send information even when it will not be used?

When information is sufficiently pervasive that it is easiest and most efficient to make it available everywhere, application programmers can use the WorkArea service to simplify programming and maintenance of code. The argument does not need to go into every argument list. It is much easier to put the value into a work area and propagate it automatically. This is especially true for methods that simply pass the value on but do nothing with it. Methods that make no use of the propagated information simply ignore it.

Work areas can hold any kind of information, and they can hold an arbitrary number of individual pieces of data, each stored as a property.

**ActivitySession Service:** The ActivitySession Service provides an alternative unit-of-work scope to that provided by global transaction contexts. An ActivitySession context can be longer-lived that a global transaction context and can encapsulate global transactions.

Although the purpose of a global transaction is to coordinate multiple resource managers, global transaction context is often used by J2EE applications as a “session” context through which to access EJB instances. An ActivitySession context is therefore a session context, and can be used in preference to a global transaction in cases where coordination of two-phase commit resource managers is not needed. Further, an ActivitySession can be associated with an HttpSession to extend a “client session” to an HTTP client.

ActivitySession support is available to Web, EJB, and J2EE-client components. EJB components can be divided into beans that exploit container-managed ActivitySessions and beans that use bean-managed ActivitySessions.

The ActivitySession service provides an API available to J2EE components that use bean-managed ActivitySessions for application-managed demarcation of ActivitySession context. The ActivitySession Service also provides a system programming interface for container-managed demarcation of ActivitySession context and for container-managed enlistment of one-phase resources in such contexts.

The UserActivitySession interface is obtained by a JNDI lookup of java:comp/beans/ActivitySession. This interface is not available to enterprise beans that use container-managed ActivitySessions, and any attempt by such beans to obtain the interface results in a NotFound exception.

**Last Participant Support:** You can coordinate the use of a single 1-phase commit (1-PC) capable resource with any number of 2-phase commit (2-PC) capable resources in the same global transaction.

At transaction commit, the 2-PC resources are prepared first using the 2-PC protocol, and, if this is successful, the 1-PC resource is then called to commit the transaction. The 2-PC commit resources are then committed or rolled back depending on the response of the 1-PC commit resource.

Coordinating access to 1-PC and 2-PC capable resources within the same transaction introduces an increased risk of a heuristic outcome to the transaction. That is, the transaction manager cannot be sure that all resources were completed in the same direction (either committed or rolled back). For this reason, to enable an application to coordinate access to 1-PC and 2-PC capable resources within the same transaction, you configure the application to accept the increased risk of a heuristic outcome.

A heuristic outcome occurs if the transaction service (JTS) receives no response from the commit 1-PC flow on the 1-PC resource. In this situation, the transaction service cannot determine whether changes for the 1-PC resource were committed or rolled back, so it cannot reliably drive the correct outcome of the global transaction on the other 2-PC resource.

You can configure the transaction service for an application server to indicate whether to log that it is about to commit the 1-PC resource. This does not reduce the heuristic hazard, but helps ensure that any failure, and subsequent recovery, of the application server during the 1-PC phase occurs with knowledge of whether or not the 1-PC resource was asked to commit.

If the 1-PC resource was asked to commit, a heuristic outcome is reported to the activity log. If the 1-PC resource was not asked to commit, then the transaction is rolled back consistently.

**CORBA Support:** The CORBA support provided in WebSphere Application Server enables the use of CORBA interfaces between the server object providing a service and a client using the service. In practice, this means that a CORBA client can access a WebSphere CORBA C++ server and a WebSphere EJ B server.

In addition, WebSphere Application Server provides a basic CORBA environment that can "bootstrap" into the J2EE name space and invoke J2EE transactions. However, it does not provide its own naming and transaction services. Therefore, a CORBA C++ client or server relies on the J2EE environment to provide these services.
**Web Services Gateway Extensions:** To extend the functionality of the Web Services Gateway, you can write your own Java programs. Although the gateway does not provide any APIs, you can use system-level interfaces. You can write a filter for the Web Services Gateway, use it to select a target service and port, capture Web service invocation information from the Web Services Gateway, and handle exceptions.

---

**Offering Information**

Program information is available through Offering Information (OITOOL) at:

http://www.ibm.com/common/ssi

and through the Passport Advantage Web site at:

http://www.ibm.com/software/passportadvantage

---

**Publications**

All publications for these programs are shipped with the electronic or CD ROM distribution.

**Displayable Softcopy Publications:** The displayable manuals are part of the basic machine-readable material. The files are shipped on the same media type as the basic machine-readable material and may be printed by the customer.

---

**Technical Information**

**Hardware Requirements:** For more information on installation requirements, visit:


WebSphere® Application Server Enterprise V5 supports the following:

- Server capable of supporting Microsoft® Windows NT® Server, V4, Service Pack 6a, or later; or Windows® 2000 Advanced Server with Service Pack 3, or later

  The server should have:
  - Intel® Pentium®, or equivalent, processor at 500 MHz, or faster
  - CD-ROM drive
  - Minimum of 190 MB available disk space for installation (without IBM SDK)
  - Minimum of 256 MB of memory, 512 MB recommended
  - Support for a communications adapter

- Server capable of running AIX V4.3.3 with the 4330-10 recommended maintenance package; or AIX® V5.1 with the 5100-02 recommended maintenance package

  The server should have:
  - RS/6000® 604e workstation at 375 MHz, or faster
  - CD-ROM drive
  - Minimum of 200 MB available disk space for installation (without IBM SDK)

- Minimum of 256 MB memory; 512 MB recommended
- Support for an appropriate network interface

- Server capable of running Red Hat Linux Advanced Server V2.1, SuSE 7.3, or SLES Linux 7.0 based on kernel 2.4

  The server should have:
  - Intel x86, or equivalent, processor at 500 MHz, or faster
  - CD-ROM drive
  - Minimum of 210 MB available disk space (without IBM SDK)
  - Minimum of 256 MB memory; 512 MB recommended
  - Support for TCP/IP and an appropriate communications adapter
  - Legacy application support RPMs installed (provided on Red Hat and SuSE distribution CDs)

- zSeries™ server capable of running distribution SLES (SuSE) 7 based on kernel 2.4

  The server should have:
  - G5, G6, or higher, processor
  - CD-ROM drive
  - Minimum of 160 MB available disk space (without IBM SDK)
  - Minimum of 256 MB memory; 512 MB recommended
  - Support for TCP/IP and an appropriate communications adapter
  - Legacy application support RPMs installed (provided on Red Hat and SuSE distribution CDs)

- Workstation running Sun Solaris operating environment, V8 at a maintenance level of July 2002, or later

  The workstation should have:
  - Sparc workstation at 440 MHz, or faster
  - CD-ROM drive
  - Minimum of 200 MB available disk space (without IBM SDK)
  - Minimum of 256 MB memory; 512 MB recommended
  - Support for TCP/IP and an appropriate communications adapter

**Software Requirements:** The following are supported software environments:

- Microsoft Windows NT Server, V4, Service Pack 6a, or later; or Windows 2000 Advanced Server with Service Pack 3, or later, for Intel, or equivalent, PCs

- AIX V4.3.3 with the 4330-10 recommended maintenance package; or AIX V5.1 with the 5100-02 recommended maintenance package for RS/6000

- Linux for Intel distribution Red Hat Linux Advanced Server V2.1, SuSE 7.3, or SuSE 7.0 based on kernel 2.4
• Linux for zSeries distribution SLES (SuSE) 7 based on kernel 2.4
• Sun Solaris operating environment, Version 8 at a maintenance level of July 2002, or later for Sun workstations

For updates to software requirements, visit:

Planning Information

Packaging: WebSphere Application Server Enterprise V5 is shipped in one package that contains the following:
• IBM International Program License Agreement (IPLA)
• Service and Support Information Sheet
• License Information (LI)
• Proof of Entitlement (PoE)

Enterprise V5 has the following number of CD-ROMs:
• Forty-eight in the program package for multiplatforms
• Forty-seven in the Passport Advantage media pack

WebSphere Application Server Enterprise for Developers V5 is shipped in one package that contains:
• IPLA
• Service and Support Information Sheet
• LI
• PoE

Enterprise for Developers V5 has the following number of CD-ROMs:
• Forty-eight in the program package for multiplatforms
• Forty-seven in the Passport Advantage media pack

This program, when downloaded from a Web site, contains the applicable IBM license agreement, and LI, if appropriate, and will be presented for acceptance at the time of installation of the program. The license and LI will be stored in a directory such as LICENSE.TXT for future reference.

Security, Auditability, and Control

WAS Enterprise V5 uses the security and auditability features of the host software.

The customer is responsible for evaluation, selection, and implementation of security features, administrative procedures, and appropriate controls in application systems and communication facilities.

Passport Advantage

Program Name

<table>
<thead>
<tr>
<th>Program Name</th>
<th>Part Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebSphere Application Server Enterprise V5 for Multiplatforms</td>
<td></td>
</tr>
<tr>
<td>Media Pack</td>
<td></td>
</tr>
<tr>
<td>LIC/SW MAINT 1 ANNI V</td>
<td>D5ALBLL</td>
</tr>
<tr>
<td>LIC/SW MAINT 2 ANNI V</td>
<td>D5ALCLL</td>
</tr>
<tr>
<td>SW MAINT AFT LIC 1 ANNI V</td>
<td>D5ALELL</td>
</tr>
<tr>
<td>SW MAINT RNWL 1 ANNI V</td>
<td>E1ALDLL</td>
</tr>
<tr>
<td>WAS Enterprise Edition FR WAS AE Processor TRD/LIC/SWMAINT 1 ANNI V</td>
<td>D5ALJLL</td>
</tr>
<tr>
<td>WebSphere Application Server Enterprise V5 for zSeries</td>
<td></td>
</tr>
<tr>
<td>Media Pack</td>
<td></td>
</tr>
<tr>
<td>LIC/SW MAINT 1 ANNI V</td>
<td>D51YVLL</td>
</tr>
<tr>
<td>LIC/SW MAINT 2 ANNI V</td>
<td>D51YWLL</td>
</tr>
<tr>
<td>SW MAINT AFT LIC 1 ANNI V</td>
<td>D51YXLL</td>
</tr>
<tr>
<td>SW MAINT RNWL 1 ANNI V</td>
<td>E00LQLL</td>
</tr>
<tr>
<td>WebSphere Application Server Enterprise V5 for Developers</td>
<td></td>
</tr>
<tr>
<td>Media Pack</td>
<td></td>
</tr>
<tr>
<td>LIC/SW MAINT 1 ANNI V</td>
<td>D5AL3LL</td>
</tr>
<tr>
<td>LIC/SW MAINT 2 ANNI V</td>
<td>D5AL4LL</td>
</tr>
<tr>
<td>SW MAINT AFT LIC 1 ANNI V</td>
<td>D5AL6LL</td>
</tr>
<tr>
<td>SW MAINT RNWL 1 ANNI V</td>
<td>E1AL5LL</td>
</tr>
</tbody>
</table>

Program Pack

| Program Pack                                                                 |             |
| WebSphere Application Server Enterprise V5 for Multiplatforms                | 45P8869     |
| WebSphere Application Server Enterprise V5 for zSeries                      | 45P9549     |
| WebSphere Application Server Enterprise V5 for Developers                   | 45P8849     |

Ordering Information

• Program Group: WebSphere Application Server
  - Program: WebSphere Application Server Enterprise V5 (5630-A37)
• Program Category: Application Server
• Program Group: WebSphere Application Server
  - Program: WebSphere Application Server Enterprise for Developers V5 (5724-D18)
• Program Category: Application Server

WebSphere Application Server Enterprise V5 is charged per processor. WebSphere Application Server Enterprise for Developers is charged per user.
Passport Advantage Customer: Media Pack Entitlement Details

Customers with active maintenance or subscription for the programs listed below are entitled to receive the corresponding media pack.

Entitled Maintenance Offerings

Description

WAS Enterprise Edition FR WAS AE Processor
TRD/LIC/SWMAINT 1 ANNV D5ALJLL

Sub-id entitlement

WASEESVR WebSphere Application Server Enterprise Edition
WASENTSB02 WebSphere Application Server Enterprise Edition for zSeries
WAPSVRUNTKIT WebSphere Application Server Enterprise Development Runtime Edition

Build to Order: Customers who have purchased a previous version of WebSphere Application Server Enterprise Edition, including maintenance, are entitled to receive WebSphere Application Server Enterprise V5 at no additional charge. Eligible customers should add the applicable CD media feature from the table below to their existing 5733-M47 or 5733-M48 record.

Feature numbers for 5733-M47 and 5733-M48 are being reused.

<table>
<thead>
<tr>
<th>Description</th>
<th>Program Number</th>
<th>Media Feature Number</th>
<th>Media</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebSphere Application Server Enterprise V5</td>
<td>5630-A37</td>
<td>5809</td>
<td>CD-ROM</td>
</tr>
<tr>
<td>1-Year Maintenance</td>
<td>5733-M47</td>
<td>5809</td>
<td>CD-ROM</td>
</tr>
<tr>
<td>3-Year Maintenance</td>
<td>5733-M48</td>
<td>5809</td>
<td>CD-ROM</td>
</tr>
</tbody>
</table>

OTC1

<table>
<thead>
<tr>
<th>Description</th>
<th>Program Number</th>
<th>Feature Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebSphere Application Server Enterprise V5.0</td>
<td>5630-A37</td>
<td>0027</td>
</tr>
<tr>
<td>Per Processor with 1 Year SW Maintenance</td>
<td></td>
<td>0031</td>
</tr>
</tbody>
</table>

1 One-time charge

This software license includes Software Maintenance, previously referred to as Software Subscription and Technical Support.

Extending coverage for a total of three years from date of acquisition may be elected. Order the program number, feature number, and quantity to extend coverage for your software licenses. If maintenance has expired, specify the after license feature number.

Description

Program Name: Maintenance IASP
PID: 1 Year 5733-M47

Use Authorizations:
Language Registration NoCharge: 2935
Multilingual
Media Choice Registration NoCharge: 5809
CD-ROM

Program Name: Maintenance IASP
PID: 3 Year 5733-M48

Use Authorizations:
Language Registration NoCharge: 2935
Multilingual
Media Choice Registration NoCharge: 5809
CD-ROM

Terms and Conditions

Licensing: IPLA. Proofs of Entitlement are required for all authorized use. Part number programs only, offered outside of Passport Advantage, where applicable, are license only and do not include Software Maintenance.

This software license includes Software Maintenance, previously referred to as Software Subscription and Technical Support.

The following agreements apply for maintenance and do not require customer signatures:

- IBM Agreement for Acquisition of Support (Z125-6011)
- Addendum for Support (Software Maintenance) for Select iSeries™ and pSeries™ Programs (Z125-6495)

LI Form Number: The LI is included with the software delivery. The LI Form Number for WebSphere Application Server Enterprise V5 is CT1N2ML. The LI Form Number for WebSphere Application Server Enterprise V5 for Developers is CT1WDML.

Limited Warranty Applies: Yes

Program Services: Available until November 30, 2005

Money-Back Guarantee: If for any reason you are dissatisfied with the program, return it within 30 days from the invoice date, to the party (either IBM or its reseller) from whom you acquired it, for a refund. This applies only to your first acquisition of the program.

Copy and Use on Home/Portable Computer: No
Volume Orders (IVO): No

Passport Advantage Applies: Yes and through the Passport Advantage Web site at:


Usage Restriction: Yes. Refer to the LI for any usage restrictions.

Software Maintenance Applies: Yes. Software Maintenance, previously referred to as Subscription and Technical Support, is now included in the Passport Advantage Agreement. Installation and technical support for the programs announced in this letter is provided by the Software Maintenance offering of the IBM International Passport Advantage Agreement. This fee service enhances customer productivity by providing voice and/or electronic access into the IBM support organizations.

IBM includes Software Maintenance with each program acquired during the coverage period. The coverage period means the period commencing on the date you acquire the program up to the first or second anniversary date, depending on whether you acquired coverage for one or two anniversaries. The coverage period for Software Maintenance acquired in the two months preceding an anniversary date will be extended to the next applicable anniversary date.

While your Software Maintenance is in effect:

IBM provides you assistance for your routine, short duration installation, usage (how-to) questions, and code-related questions. IBM provides assistance via telephone and, if available, electronic access, only to your IS technical support personnel during the normal business hours (published prime shift hours) of your IBM support center. This assistance is not available to your end users.

IBM provides Severity 1 assistance 24 hours a day, every day of the year. For additional details, visit:


Software Maintenance does not include assistance for:

- The design and development of applications
- Your use of programs in other than their specified operating environment
- Failures caused by programs for which IBM is not responsible under this agreement

For more information, about the Passport Advantage Agreement, refer to the IBM International Passport Advantage Agreement Software Announcement 201-202, dated July 10, 2001, or visit:


All distributed software licenses include Software Maintenance (software subscription and technical support) for a period of 12 months from the date of acquisition providing a streamlined way to acquire IBM software and ensure technical support coverage for all licenses. Extending coverage for a total of three years from date of acquisition may be elected. For additional information, refer to Software Announcement 201-201, dated July 10, 2001.

AIX/UNIX® Upgrade Protection Applies: No

Entitled Upgrade for Current AIX/UNIX Upgrade Protection Licensees: No

Variable Charges Apply: No

Educational Allowance Available: Yes, 15% education allowance applies to qualified education institution customers.

<table>
<thead>
<tr>
<th>Program Name</th>
<th>Part Number</th>
<th>OTC</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebSphere Application Server Enterprise V5 for Multiplatforms Program Pack</td>
<td>45P8869</td>
<td>$25,000</td>
</tr>
<tr>
<td>WebSphere Application Server Enterprise V5 for zSeries Program Pack</td>
<td>45P9549</td>
<td>25,000</td>
</tr>
<tr>
<td>WebSphere Application Server Enterprise Developer V5 Program Pack</td>
<td>45P8849</td>
<td>2,999</td>
</tr>
</tbody>
</table>

For Passport Advantage and charges, contact your authorized Lotus® Business Partner. Additional information is also available on the Passport Advantage at:


Program Name: WebSphere Application Server Enterprise V5 Program Number: 5630-A37

<table>
<thead>
<tr>
<th>Description</th>
<th>Feature Number</th>
<th>OTC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Per Processor with 1-Year SW Maintenance</td>
<td>0027</td>
<td>$27,036</td>
</tr>
</tbody>
</table>

Description | Feature Number | Basic Upgrade Charge |
-------------|----------------|---------------------|
Per Processor with 1-Year SW Maintenance (Upgrade from WebSphere Application Server Network Deployment) | 0031 | $13,518 |

Global Financing

IBM Global Financing offers competitive financing to credit-qualified customers to assist them in acquiring IT solutions. Offerings include financing for IT acquisition, including hardware, software and services, both from IBM and other manufacturers or vendors. Offerings (for all customer segments: small, medium and large enterprise), rates, terms, and availability can vary by country. Contact your local IBM Global Financing organization or visit:

Order Now

To order, contact the Americas Call Centers, your local IBM representative, or your IBM Business Partner.

To identify your local IBM representative or IBM Business Partner, call 800-IBM-4YOU (426-4968).

Phone: 800-IBM-CALL (426-2255)
Fax: 800-2IBM-FAX (242-6329)
Internet: ibm_direct@vnet.ibm.com
Mail: The Americas Call Centers
Dept. YE001
P.O. Box 2690
Atlanta, GA 30301-2690

Reference: YE001

The Americas Call Centers, our national direct marketing organization, can add your name to the mailing list for catalogs of IBM products.

Note: Shipments will begin after the planned availability date.

Trademarks

zSeries, iSeries, and pSeries are trademarks of International Business Machines Corporation in the United States or other countries or both.
WebSphere, AIX, and RS/6000 are registered trademarks of International Business Machines Corporation in the United States or other countries or both.
Intel and Pentium are registered trademarks of Intel Corporation.
Microsoft is a trademark of Microsoft Corporation.
Windows NT and Windows are registered trademarks of Microsoft Corporation.
UNIX is a registered trademark of the Open Company in the United States and other countries.
Lotus is a registered trademark of Lotus Development Corporation and/or IBM Corporation.
Other company, product, and service names may be trademarks or service marks of others.