Redpaper

Alan Corcoran
Balazs Csepregi-Horvath
Addison Goering
Jose Pablo Hernandez
Julien Limodin
Sergio Pinto

WebSphere Application Server V8.0 Technical Overview

IBM WebSphere® Application Server is the implementation by IBM® of the Java Platform, Enterprise Edition (Java EE) platform. It conforms to the Java EE 6 specifications as one of its supporting programming models. WebSphere Application Server is available in unique packages that are designed to meet a wide range of customer requirements. At the heart of each package is a WebSphere Application Server that provides the runtime environment for enterprise applications.

This IBM Redpaper™ publication discusses the runtime server component of WebSphere Application Server.

For more information about topics that we discuss in this paper, refer to *WebSphere Application Server V8: Concepts, Planning, and Design*, SG24-7957.

New features in Version 8

This section summarizes the new support in WebSphere Application Server V8. This list is based on WebSphere Application Server Network Deployment for all operating systems. More information about each of these features is in the WebSphere Application Server Information Center article, *What is new in this release* at:

http://publib.boulder.ibm.com/infocenter/wasinfo/v8r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/welc_newinrelease.html

Programming model, standards, and tooling support:

- Java Batch programming model: Build batch applications to perform long running bulk transaction processing and computationally intensive work.
- Communications Enabled Applications (CEA) programming model: Enable existing applications to quickly take advantage of communications features that involve phone calls and web collaboration.
- Extensible Markup Language (XML) programming model: Build web applications that process data using standard XML technology, including Extensible Stylesheet Language (XSLT 2.0), XML Path Language (XPath 2.0), and XML Query Language (XQuery 1.0).
- Java Enterprise Edition 6.0 support, including:
  - Enterprise JavaBeans (EJB) 3.1
  - Contexts and Dependency Injection (CDI) 1.0
  - Java Connector Architecture (JCA) Version 1.6
  - Java Persistence API (JPA) 2.0
  - JavaServer Faces (JSF) 2.0
  - JavaServer Pages (JSP) 2.2
  - Unified Expression Language (EL) 2.2
  - Java Servlet 3.0
  - Web Services for Java EE (JSR 109) Version 1.3
  - JAX-WS Version 2.2
  - JAXB Version 2.2
  - Java API for RESTful Web Services (JAX-RS) 1.1
  - HTTP servlets profile aspects of JSR 196: Java Authentication SPI for Containers (JASPI) specification
  - java:global, java:app, and java:module Java Naming and Directory Interface (JNDI) names for applications deployed on one server.
  - Bean Validation 1.0
- OSGi applications programming model: Build and deploy modular applications that use both Java EE and OSGi technologies. Use OSGi Enterprise Specification 4.2 Blueprint Container for declarative assembly of components.
- Service Component Architecture (SCA) programming model: Build applications tailored for delivery and management in service-oriented architecture (SOA) environments. Derived from open SOA Collaboration (osoa.org) SCA 1.0 specifications.

- Session Initiation Protocol (SIP) programming model: Build converged communication-enhanced applications. Includes support for SIP Servlet Specification 1.1, which is also referred to as Java™ Specification Request (JSR) 289.
- New and improved sample applications to help you learn about the technologies in the product: Includes SCA, CEA, OSGi applications, XML, and the Internationalization service.

Integrated and optimized developer tooling, Rational® Application Developer Version 8.0.3, provides capabilities for you to develop applications, assemble, and deploy your applications to WebSphere Application Server Version 8.0. It also includes a rapid-deployment feature for testing applications in the Version 8.0 runtime environment.

Product and maintenance installation support for administrators:

- Simplified installation and uninstallation for the product and maintenance using the IBM Installation Manager.
- Install and apply maintenance on remote computers using centralized installation manager (CIM). You can use CIM to install and manage Installation Manager instances. CIM functions are integrated with the job manager.

Improved productivity:

- Numerous improvements to the administration tools for ease of administrative tasks, including jobs management, application management, and diagnostic activity.
- Resource workload routing includes data source and connection factory fail over and subsequent fail back from a predefined alternate or backup resource.
- Support for connecting to multi-instance WebSphere MQ queue managers.
- A variety of programming support features to improve productivity.
- Install, uninstall, and update enterprise application files by adding them to a monitored directory (Distributed and z/OS systems).
- Additional database support. WebSphere Application Server Version 8.0 was tested with the following new databases and JDBC drivers:
  - Derby 10.5
  - DataDirect Connect for Java Database Connectivity (JDBC) driver V4.2
  - Microsoft® SQL Server Enterprise 2008 R2
  - Microsoft SQL Server JDBC Driver, version 3.0
  - Oracle 11g Standard Edition and Enterprise Release 2
  - Oracle 11g Release 2 JDBC Driver

Improved security features:

- Numerous additional support features for SAML, Web services, security, LTPA, and single sign-on.
- Support for JSR 196: Java Authentication SPI for Containers (JASPI) specification.
- Support for configuring federated repositories at the domain level in a multiple security domain environment.
- Support for z/OS System Authorization Facility (SAF) security to associate a SAF user ID with a distributed identity.
- Support for all security updates as defined in the Java Servlet 3.0 specification (JSR-315), including the new servlet security annotations, use of new programmatic security APIs, and the dynamic updating of the servlet security configuration.

- Enhancements to security auditing and reporting.
- More security features for the server are enabled by default for enhanced security hardening including session security and requiring SSL for IIOP communications.

High performance features:

- High availability connections to a WebSphere MQ queue manager over an MQ link.
- The ability to disable WebSphere MQ functionality in WebSphere Application Server, provided when it is not required.
- Cutting-edge performance for real applications in commercial environments.
- A unified clustering framework that brings workload balancing, dependability, and other benefits to the application servers in your heterogeneous environment.

WebSphere Application Server packaging

WebSphere Application Server comes in several packaging options. In addition to the application server component, each package contains an appropriate combination of complementary products, for example, IBM HTTP Server, IBM Assembly and Deploy Tools for WebSphere Administration, Edge components, and other products.

Distributed platforms

WebSphere Application Server V8.0 has the following packaging options for distributed platforms, including IBM AIX®, HP-UX, Linux, Solaris, and Microsoft Windows:

- IBM WebSphere Application Server Express V8.0, referred to as *Express*

  For more information about Express, see:

  http://www.ibm.com/software/webservers/appserv/express/

- IBM WebSphere Application Server V8.0, referred to as *Base*

  For more information about Base, see:

  http://www.ibm.com/software/webservers/appserv/was/

- IBM WebSphere Application Server Network Deployment V8.0, referred to as *Network Deployment*

  For more information about Network Deployment, see:

  http://www.ibm.com/software/webservers/appserv/was/network/

System z

For WebSphere Application Server on IBM System z® and IBM WebSphere Application Server for z/OS® V8.0, a full-function version of the Network Deployment product is available.

For more information about WebSphere Application Server for z/OS V8.0, see:

http://www.ibm.com/software/webservers/appserv/zos_os390/

System i

WebSphere Application Server on IBM System i® has the following packaging options:

- IBM WebSphere Application Server V8.0 for IBM i
- IBM WebSphere Application Server for Developers V8.0 for IBM i
- IBM WebSphere Application Server Network Deployment V8.0 for IBM i
- IBM WebSphere Application Server - Express V8.0 for IBM i

For more information about WebSphere Application Server on System i, see:

http://www.ibm.com/systems/i/software/websphere/index.html

Development

WebSphere Application Server for Developers V8.0 is supported on IBM AIX, IBM i, HP-UX, Linux, Solaris, and Microsoft Windows. It is available as a download with support available for purchase. For more information about WebSphere Application Server for Developers V8.0, see:

http://www.ibm.com/software/webservers/appserv/developer/index.html

Application support

WebSphere Application Server V8.0 can run the following types of applications:

- Java EE applications
- Portlet applications
- Session Initiation Protocol (SIP) applications
- Business level applications
- Java Batch applications
- OSGi applications
- Communications enabled applications (CEA)
- Service Component Architecture (SCA)
- XML

Java EE applications

The Java EE specification is the standard for developing, deploying, and running enterprise applications. WebSphere Application Server V8.0 provides full support for the Java EE 6 specification. The Java EE programming model has multiple types of application components:

- Enterprise beans
- Servlets and JavaServer Pages
- Application clients

The primary development tools for WebSphere Application Server Java EE 6 applications are:

- IBM Rational® Application Developer Standard Edition for WebSphere V8
- IBM Rational Application Developer for WebSphere V8

A tool is included in this version called IBM Assembly and Deploy Tools for WebSphere Administration. This tool replaces the previously available IBM Rational Application Developer Assembly and Deploy Tool. It allows the developer to accomplish key assembly and deployment needs, including editing deployment artifacts, developing and testing scripts, and deploying and debugging applications. It is not intended for general application development.

For information about the Java EE specification, see:

http://java.sun.com

Portlet applications

The Portlet container in WebSphere Application Server V8.0 provides the runtime environment for Java Specification Request (JSR) 286 compliant portlets. Portlet applications are intended to be combined with other portlets collectively to create a single page of output. The primary development tool for portlets on WebSphere Application Server portlet applications is Rational Application Developer for WebSphere V8.

Portlets are packaged in web archive (WAR) files. The portlet run time does not provide the advanced capabilities of WebSphere Portal, such as portlet aggregation and page layout, personalization and member services, or collaboration features.

For more information about JSR 286, see:

http://jcp.org/en/jsr/detail?id=286

Communications enabled applications

Communications enabled applications (CEA) allow developers to add dynamic web communications to any application or business process. CEA provides Representational State Transfer (REST) and web service interfaces to enable existing applications to take advantage of communication features involving phone calls and web collaboration.

CEA applications do not require developers to have extensive knowledge of telephony or Session Initiation Protocol (SIP) to implement this type of applications. CEA capabilities deliver call control, notifications, and interactivity, providing the platform for more complex communications.

Session Initiation Protocol applications

SIP applications are Java programs that use at least one SIP servlet written to the JSR 116 specification. WebSphere Application Server V8.0 also supports SIP Servlet Specification 1.1, also referred to as *JSR 289*. SIP is used to establish, modify, and terminate multimedia IP sessions. SIP negotiates the medium, the transport, and the encoding for the call. After the SIP call is established, the communication takes place over the specified transport mechanism, independent of SIP. Examples of application types that use SIP include voice over IP (VOIP), click-to-call, and instant messaging.

Rational Application Developer for WebSphere V8 provides special tools for developing SIP applications. SIP applications are packaged as SIP archive (SAR) files, and are deployed to the application server using the standard WebSphere Application Server administrative tools. SAR files can also be bundled in a Java EE application archive (EAR file), similar to other Java EE components.

In the application server, the web container and SIP container are converged and can share session management, security and other attributes. In this model, an application that includes SIP servlets, HTTP servlets, and portlets can interact, regardless of the protocol. High availability of these converged applications is made possible because of the integration of HTTP and SIP in the base application server.

For more information about SIP applications, see the following resources:

- JSR 289 SIP Servlet API 1.1 Specification

  http://www.jcp.org/aboutJava/communityprocess/final/jsr289/index.hml
- JSR 116

  http://jcp.org/en/jsr/detail?id=116
- RFT 3261

  http://www.ietf.org/rfc/rfc3261.txt

Business level applications

A *business level application* is a notion of an application beyond Java EE's definition. This concept is an administration concept that expands the options previously offered by Java EE. The grouping notion for enterprise-level applications includes WebSphere and non-WebSphere artifacts, such as Service Component Architecture (SCA) packages, libraries, and proxy filters under a single application definition, as illustrated in Figure 1. Every artifact in the group is a *composition unit*.

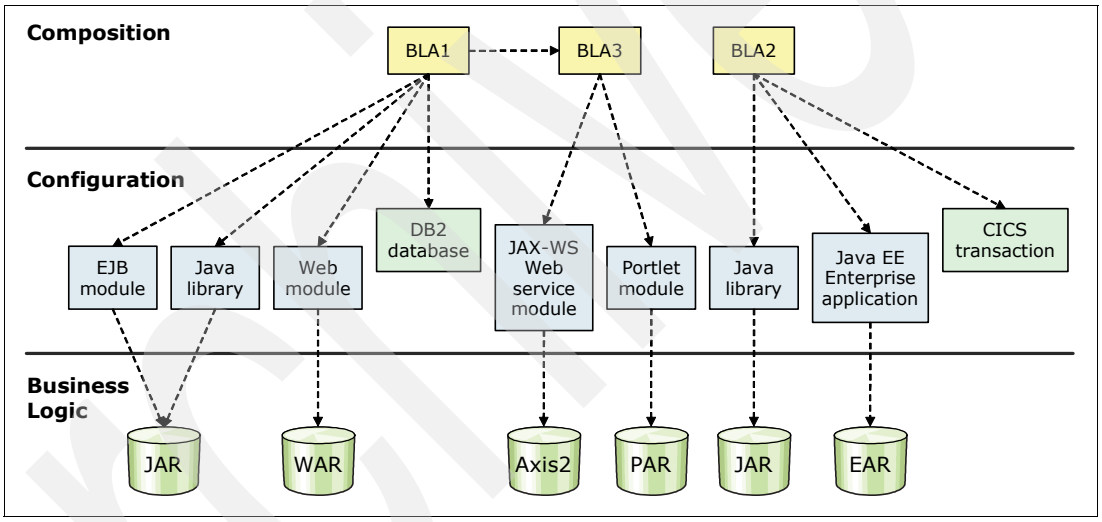

Figure 1 Business level applications

A business level application can be useful when an application has the following characteristics:

- Is composed of multiple packages
- Applies to the post-deployment side of the application life cycle
- Contains additional libraries, or non-Java EE artifacts
- Includes artifacts that run on heterogeneous environments that include WebSphere and non-WebSphere runtimes
- Is defined in a recursive manner, for example, if an application includes other applications

Java batch applications

Batch applications can perform long running bulk transaction processing and computationally intensive work. They run as background jobs, described by a job control language, and are supported by infrastructure components that are aimed to support batch workloads.

The control language is called *XML job control language (xJCL)*. It allows users to describe the job steps that are involved in their batch jobs. This application runs in batch containers that also run at the same time in designated WebSphere Application Server environments. A job scheduler determines the appropriate container to run the job and maintains job histories to support job visibility and control.

OSGi applications

OSGi applications are modular applications that use both Java EE and OSGi technologies. With this design, you can improve control and flexibility by designing and building applications and groups of applications from coherent, multiversion, and reusable OSGi bundles.

WebSphere Application Server V8.0 support for OSGi applications includes the capability of deploying web applications that use the Java Servlet 3.0 Specification. Additionally, you can update a running application and impact only those bundles that are affected by the update. This method allows you to extend and scale running applications as business demands it.

OSGi applications allow the composition of isolated enterprise applications using multiple, multiversion bundles that have dynamic life cycles. Application maintenance and upgrades can be simplified using standard OSGi mechanisms to simultaneously load multiple versions of classes in the same application. Existing web archives (WAR files) can be reused and deployed as OSGi web application bundles.

For more information about OSGi applications, refer to:

`http://www.osgi.org/About/WhatIsOSGi`

Application server configurations

At the core of each product in the WebSphere Application Server family is an application server. Three product members of the WebSphere Application Server family are discussed in this book:

- IBM WebSphere Application Server Express V8.0, referred to as *Express*
- IBM WebSphere Application Server V8.0, referred to as *Base*
- IBM WebSphere Application Server Network Deployment V8.0, referred to as *Network Deployment*

Each member has essentially the same architectural structure shown in Figure 2 on page 9. The application server structure for the Base and Express platforms is identical. Licensing terms and platform support differentiate the products.

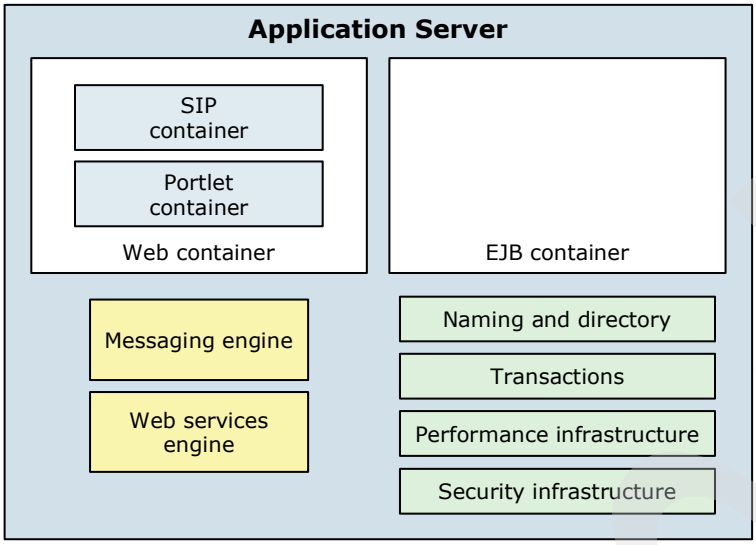

Figure 2 Base and Express architecture

The Base and Express platforms are limited to stand-alone application servers. Each application server can be administered individually or multiple application servers can be managed through the administrative agent. In an Express installation, there is no workload management or high availability capabilities among application servers. Each application server operates individually. This is the default configuration with Base; however, in a Base installation, you can configure simple load balancing across multiple stand-alone application servers at the web tier using the web server plug-in that is provided with WebSphere Application Server. The number of application servers that you can include in this configuration is bound by the limits that exist in the WebSphere Application Server license agreement. The Network Deployment platform enables more advanced topologies that provide workload management, scalability, high availability, and central management of multiple application servers, as shown in Figure 3 on page 10.

Runtime environments are built by creating profiles. A profile can define a deployment manager, a stand-alone application server or an empty node to be federated (added) to a cell. Each profile contains files that are specific to that run time, such as logs and configuration files. Profiles can be created during and after installation. After the profiles are created, further configuration and administration is performed using the WebSphere administrative tools.

Stand-alone application servers

All WebSphere Application Server packages support a single stand-alone server environment, as illustrated in Figure 3 on page 10. With this configuration, each application server acts as an unique entity. An application server runs one or more applications and provides the services that are required to run those applications. Each stand-alone server is created by defining an application server profile.

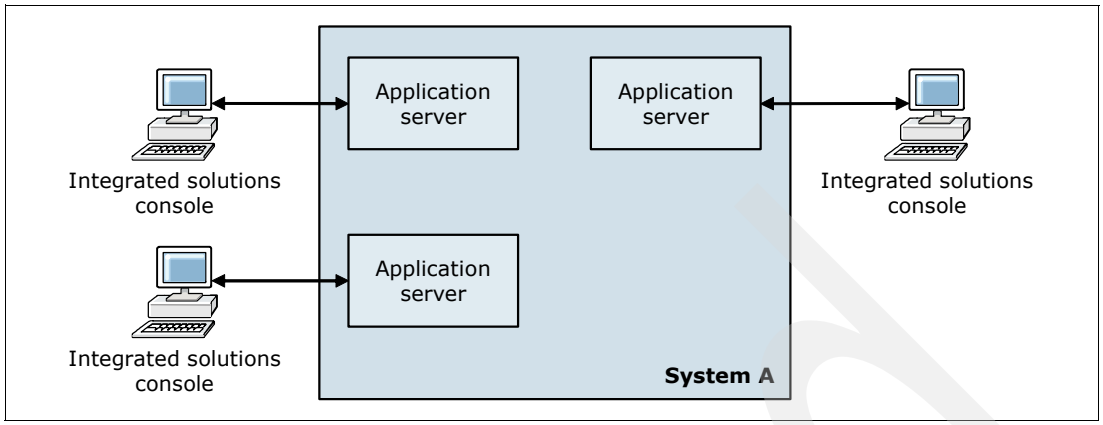

Figure 3 Stand-alone application server configuration

Multiple stand-alone application servers can exist on a system, either through independent installations of the WebSphere Application Server product binaries or by creating multiple application server profiles within one installation. The base edition of WebSphere Application Server supports simple failover of up to five application server instances. This is done by combining the plug-in configuration files of multiple stand-alone application server profiles into a single configuration file.

WebSphere Application Server for z/OS provides workload balancing and response time goals on a transactional base and a special clustering mechanism, the multi-servant region, with a stand-alone application server.

Distributed application servers

With the Network Deployment packaging, you can build a distributed server configuration to enable central administration, workload management, and failover. In this environment, you integrate one or more application servers into a cell that is managed by a central administration instance, a deployment manager, which we explain in “Deployment managers” on page 14. The application servers can reside on the same system as the deployment manager or on multiple separate systems. Administration and management is handled centrally from the administration interfaces of the deployment manager, as shown in Figure 4.

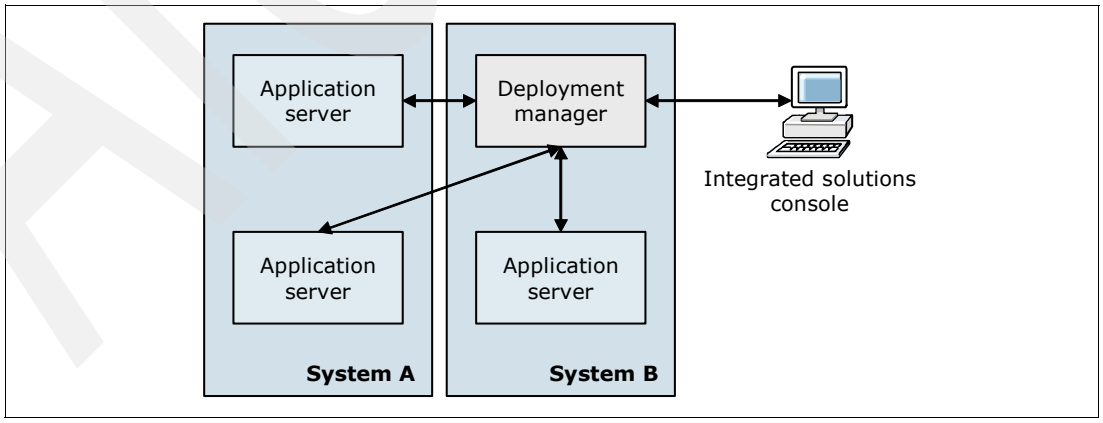

Figure 4 Distributed application servers with WebSphere Application Server V8

With a distributed server configuration, you can create multiple application servers to run unique sets of applications and manage those applications from a central location. However, more importantly, you can cluster application servers to allow for workload management and

failover capabilities. Applications installed in the cluster are replicated throughout the application servers. The cluster can be configured so that when one server fails, another server in the cluster continues processing.

Workload is distributed among web and Enterprise JavaBeans (EJB) containers in a cluster using a weighted round-robin scheme. In z/OS, the weighted round-robin mechanism is replaced by the integration of WebSphere Application Server for z/OS in the Workload Manager (WLM). The WLM is an integral part of the operating system that allows requests to be dispatched to a cluster member according to real-time load and whether the member reaches its defined response time goals.

By creating various profiles, you can create a distributed server configuration using one of the following methods:

- Create a deployment manager profile to define the deployment manager. Then, create one or more custom node profiles. The nodes that are defined by each custom profile can be federated into the cell that is managed by the deployment manager during profile creation or manually later. The custom nodes can exist inside the same operating system image as the deployment manager or in another operating system instance. You can then create application servers using the Integrated Solutions Console, **`wsadmin`** scripts, or in interactive wsadmin mode.

  This method is useful when you want to create multiple nodes, multiple application servers on a node, or clusters.

- Create a deployment manager profile to define the deployment manager. Next, create one or more application server profiles and federate these profiles into the cell that is managed by the deployment manager. This process adds both nodes and application servers into the cell. The application server profiles can exist on the deployment manager system, on multiple separate systems, or a z/OS images.

  This method is useful in development or small configurations. Creating an application server profile provides the option of having the sample applications installed on the server. When you federate the server and node to the cell, any installed applications can be carried into the cell with the server.

- Create a cell profile. This method actually creates two profiles, a deployment manager profile and a federated application server profile. Both profiles reside on the same system.

  This method is useful in a development or test environment. Creating a single profile provides a simple distributed system on a single server or z/OS images.

Application servers concepts

Regardless of the configuration, WebSphere Application Server is organized based on the concept of *cells*, *nodes*, and *servers*. Although all of these elements are present in each configuration, cells and nodes do not play an important role until you take advantage of the features that are provided with Network Deployment. These cells, nodes, and servers can be managed by a combination of deployment managers, administrative agents, and job managers.

Application servers

The *application server* is the primary runtime component in all configurations and is where an application actually executes. All WebSphere Application Server configurations can have one or more application servers. In the Express and Base configurations, each application server functions as a separate entity. There is no central administration among application servers.

The base edition of WebSphere Application Server supports simple failover of up to five application server instances. With Network Deployment, you can build a distributed server environment consisting of multiple application servers maintained from a central administration point. In a distributed server environment, you can cluster application servers for workload distribution and failover.

Nodes, node groups, and node agents

This section defines node-related concepts.

Nodes

A *node* is an administrative grouping of application servers for configuration and operational management within one operating system instance (virtualization allows multiple operating systems on one machine). It is possible to create multiple nodes inside one operating system instance, but a node cannot leave the operating system boundaries. In a stand-alone application server configuration, there is only one node. With Network Deployment, you can configure a distributed server environment that consists of multiple nodes that are managed from one central administration server. See Figure 5.

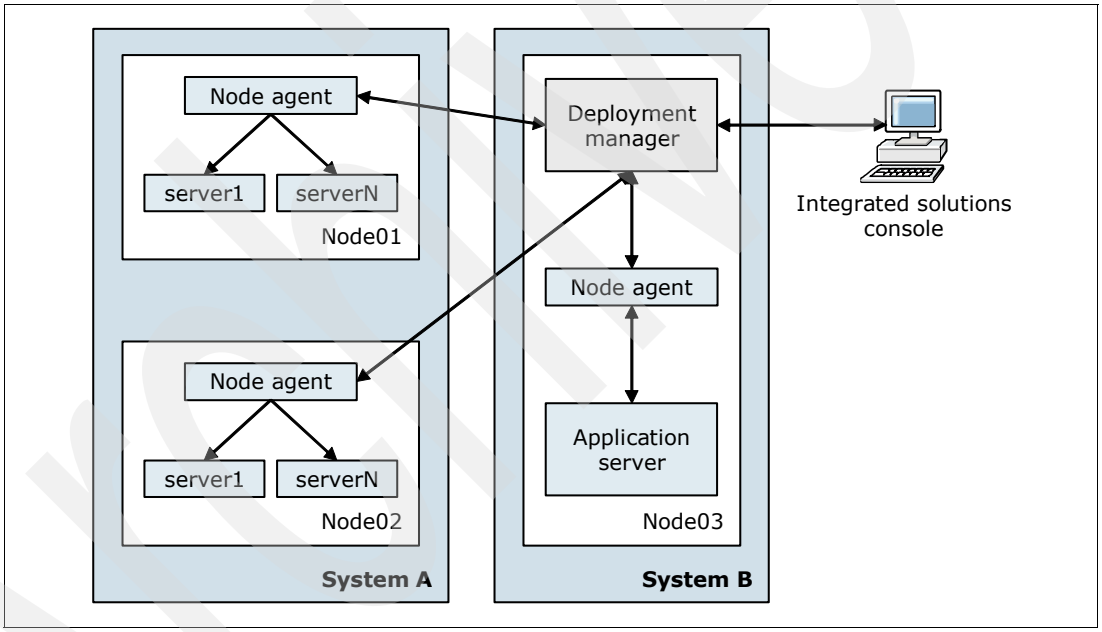

Figure 5 Node concept in a WebSphere Application Server network deployment configuration

Node agents

In distributed server configurations, each node has a *node agent* that works with the deployment manager to manage administration processes, as shown in Figure 5. A node agent is created automatically when you add (federate) a stand-alone node to a cell. It is not included in Base and Express configurations.

Node groups

A *node group* is a grouping of nodes within a cell that have similar capabilities. A node group validates that the node can perform certain functions before allowing them. For example, a cluster cannot contain both z/OS nodes and nodes that are not z/OS-based.

In this case, you can define multiple node groups:

- One group for the z/OS nodes
- One group for nodes other than z/OS

A DefaultNodeGroup is created automatically. This node group contains the deployment manager and any new nodes with the same platform type. A node can be a member of more than one node group.

On the z/OS platform, a node must be a member of a *system complex* (sysplex) node group. Nodes in the same sysplex must be in the same sysplex node group. A node can be only in one sysplex node group.

Cells

A *cell* is a grouping of nodes into a single administrative domain. In the Base and Express configurations, a cell contains one node. That node contains one server. See Figure 6.

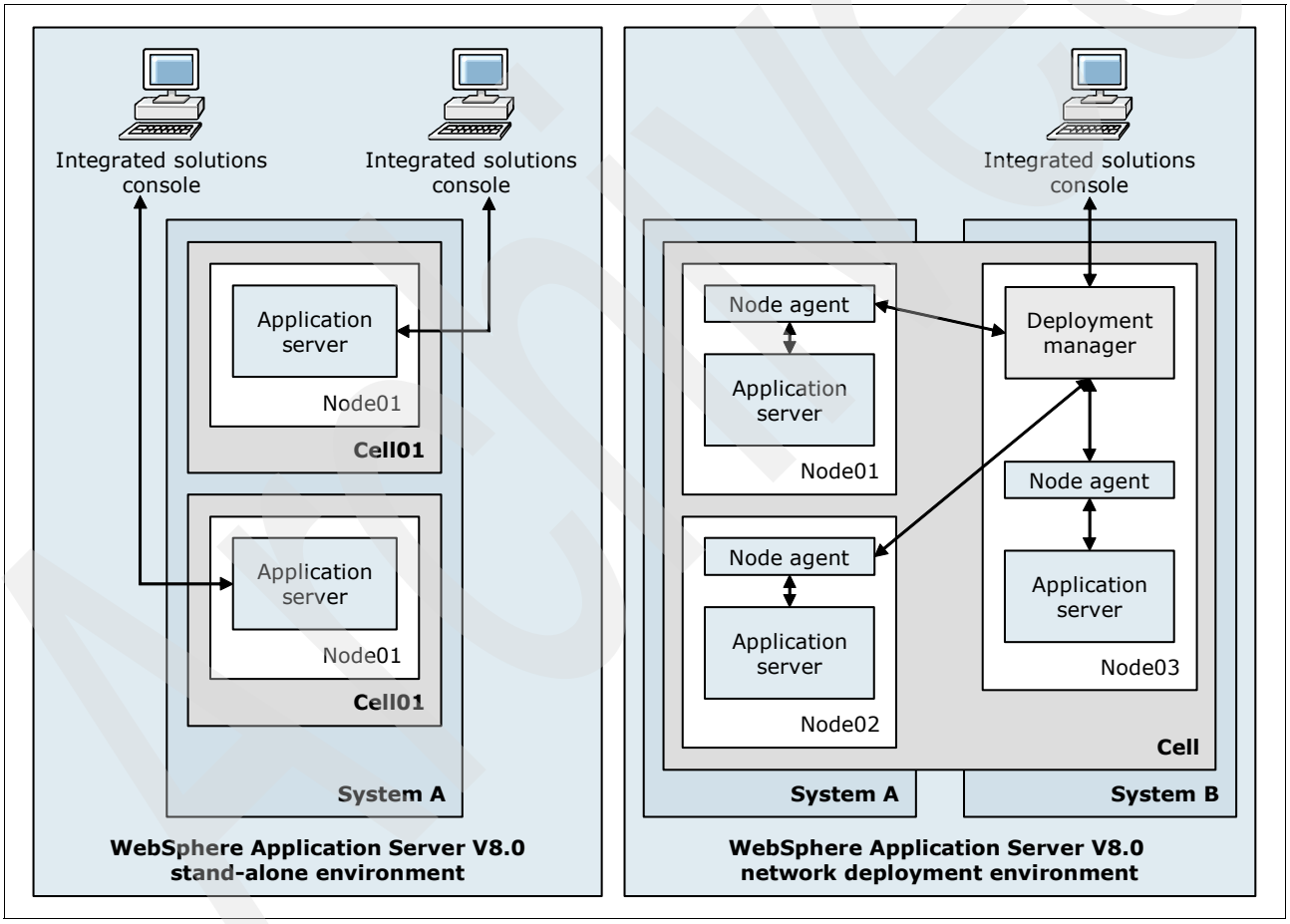

Figure 6 Cells in a WebSphere Application Server topology

In a Network Deployment environment, a cell can consist of multiple nodes (and node groups), which are all administered from a single point, the deployment manager. See Figure 6. If your cell configuration contains nodes running on the same platform, it is called a *homogeneous cell*. It is also possible to have a cell made up of nodes on mixed platforms. This is referred to as a *heterogeneous cell*.

Deployment managers

The *deployment manager* is the central administration point of a cell, which consists of multiple nodes and node groups in a distributed server configuration. The deployment manager uses the node agent to manage the application servers within one node.

A deployment manager provides management capability for multiple federated nodes and can manage nodes that span multiple systems and platforms. A node can only be managed by a single deployment manager, and must be federated to the cell of that deployment manager.

The configuration and application files for all nodes in the cell are centralized into a master configuration repository. This centralized repository is managed by the deployment manager and synchronized with local copies that are held on each of the nodes. The entire centralized repository is not synchronized to each node. Only the node's subset of the repository is replicated. See Figure 7.

The deployment manager can also send asynchronous jobs to unfederated nodes using the Job Manager in the administrative console.

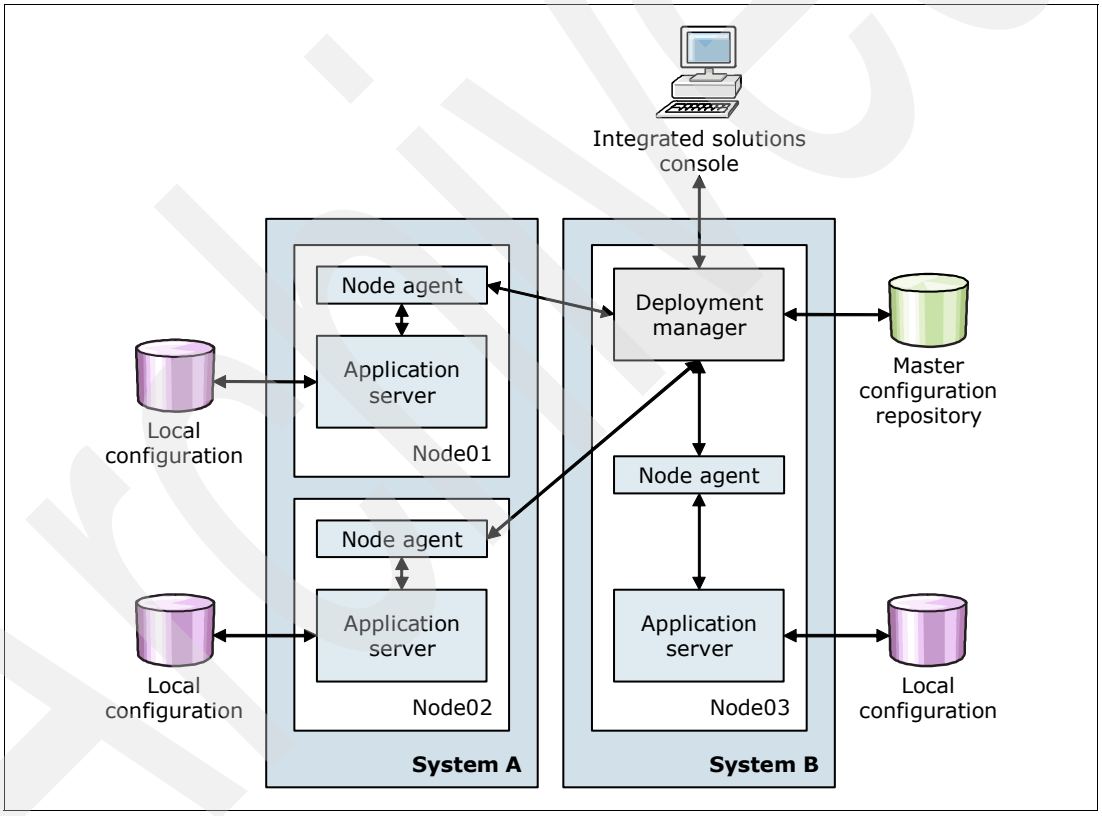

Figure 7 Configuration repositories in a network deployment installation

Administrative agents

An *administrative agent* is a component that provides enhanced management capabilities for stand-alone (Express and Base) application servers, as illustrated in Figure 8. This concept was first introduced with WebSphere Application Server V7.0 to separate the administrative components from the application server run time.

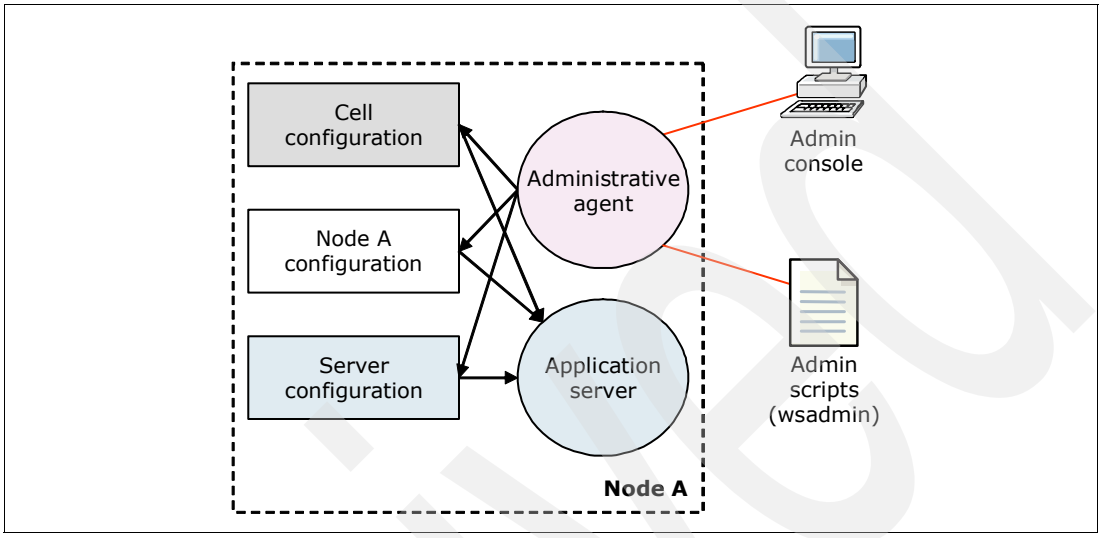

Figure 8 Administrative agent in Express and Base configurations

All configurations that are related to the application server are directly connected to the administrative agent that provides services to administrative tools. An administrative agent can manage multiple stand-alone server instances on a single system or z/OS image. Therefore, when using an administrative agent, as the number of application server instances increases, the redundancy of the administration footprint (for each application server) is eliminated.

Job managers

A *job manager* is a component that provides management capabilities for multiple stand-alone application servers, administrative agents, and deployment managers, as illustrated in Figure 9. It brings enhanced multiple node installation options for your environment.

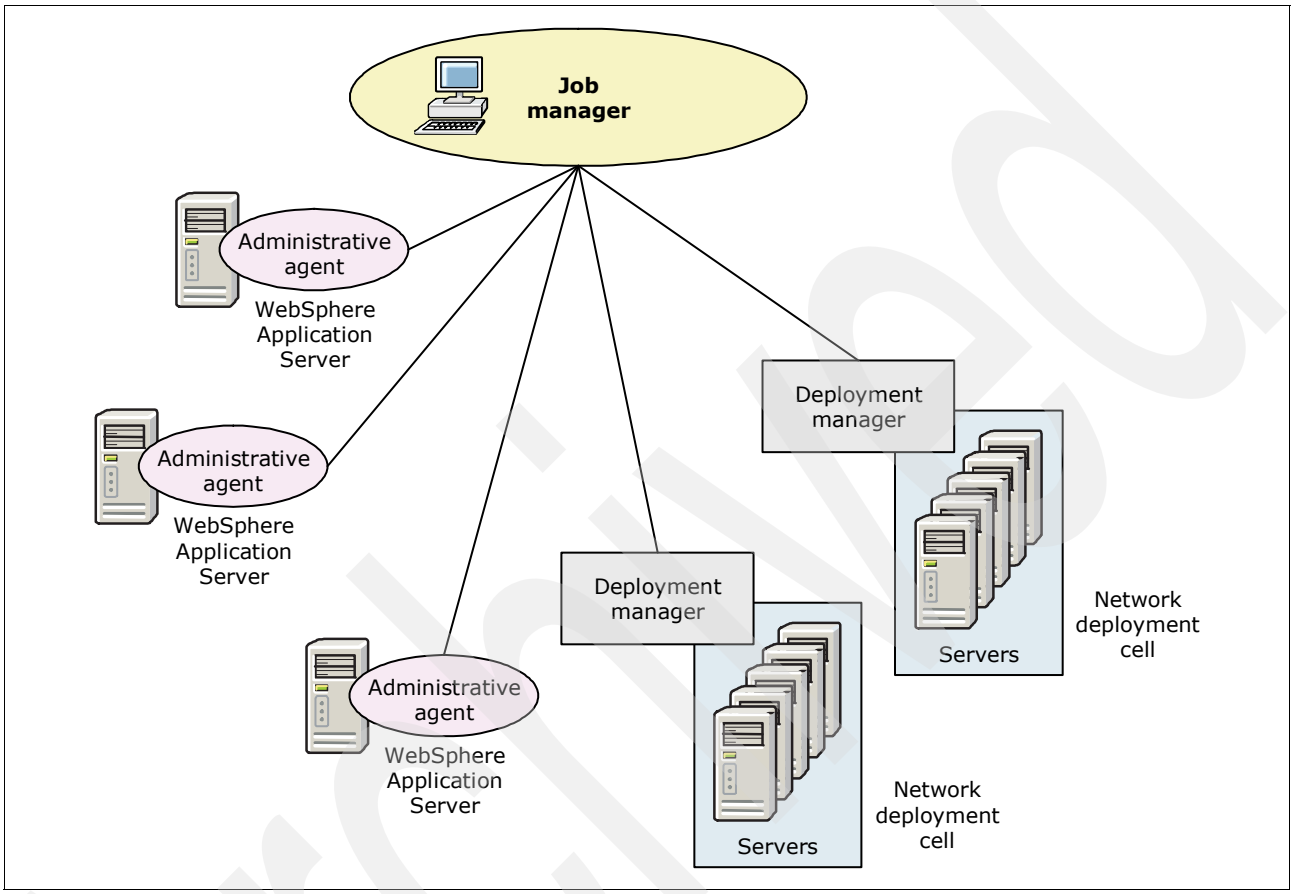

Figure 9 High-level overview of the job manager architecture

There are many benefits of the job manager to execute daily tasks in one step for multiple installations. This includes starting and stopping servers, distributing and deploying applications, and various other actions. These jobs can be submitted and executed asynchronously.

A new capability in the job manager in WebSphere Application Server V8 is the ability to submit Centralized Installation Manager (CIM) jobs using the job managers to multiple cells and server topologies.

The job manager was first introduced with WebSphere Application Server V7.0 and is available only with WebSphere Application Server Network Deployment and WebSphere Application Server for z/OS.

Servers

WebSphere Application Server provides the infrastructure and capabilities that are required to host applications and proxy servers that distribute work to the application servers. It also provides the ability to define external servers to the administration process and to associate web servers for routing purposes.

Application servers

Application servers provide the runtime environment for application code. They provide containers and services that specialize in enabling the execution of specific Java application components. Each application server runs in its own Java Virtual Machine (JVM).

WebSphere Application Server V8 has an enhanced JVM designed to improve stability and performance. It provides a Java language compiler and execution environment to support the Java Platform, Standard Edition (Java SE) 6 specification. This JVM is supported on all platforms that ship with an IBM JDK.

Application server clusters

An *application server cluster* is a logical collection of application server processes that provides workload balancing and high availability. It is a grouping of application servers that run an identical set of applications that are managed so that they behave as a single application server (parallel processing). WebSphere Application Server Network Deployment or WebSphere Application Server for z/OS is required for clustering.

Application servers that are a part of a cluster are called *cluster members*. When you install, update, or delete an application, the updates are automatically distributed to all members in the cluster. A rollout update option enables you to update and restart the application servers on each node, one node at a time, providing continuous availability of the application to the user.

Proxy servers

A *proxy server* is a specific type of application server that routes requests to content servers that perform the work. The proxy server is the initial point of entry, after the protocol firewall, for requests entering the environment.

WebSphere Application Server allows you to create two types of proxy servers:

- WebSphere Application Server Proxy

  This proxy server is used to classify, prioritize, and route HTTP and SIP requests to servers in the enterprise and to cache content from servers.

- DMZ Secure Proxy Server

  This proxy server comes in a separate installation package and provides security enhancements to allow deployments inside of a demilitarized zone.

Web servers

Although web servers are independent products, they can be defined to and managed by the WebSphere Application Server administration process. The primary purpose for this is to enable the administrator to associate applications with one or more defined web servers to generate the proper routing information for web server plug-ins if multiple servers are used.

Web servers are associated with two types of nodes:

- *Managed* nodes have a node agent on the web server system that allows the deployment manager to administer the web server. You can start or stop the web server from the deployment manager, generate the web server plug-in for the node, and automatically push it to the web server. In most installations, you have managed web server nodes behind the firewall with the WebSphere Application Server installations. See Figure 10.

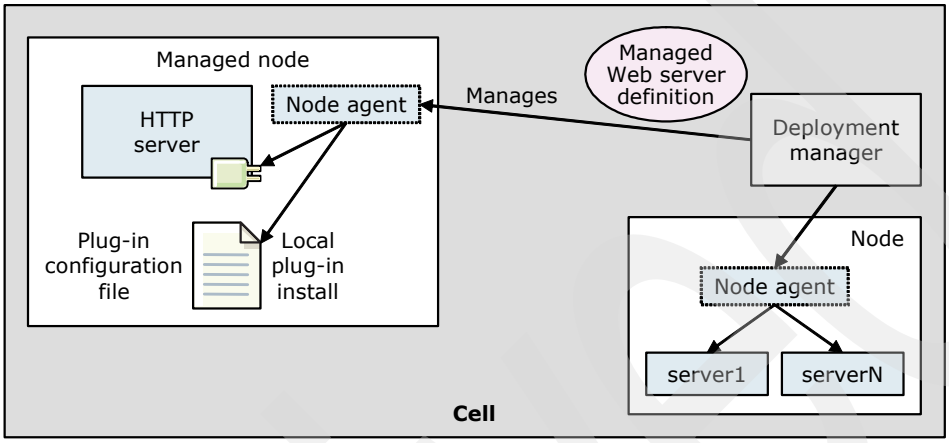

Figure 10 Web server on a managed node

- *Unmanaged* nodes, Figure 11, are not managed by WebSphere. You usually find these outside the firewall or in the demilitarized zone. You must manually transfer the web server plug-in configuration file to the web server on an unmanaged node. In a z/OS environment, you must use unmanaged nodes if the web server is a not running on the z/OS platform.

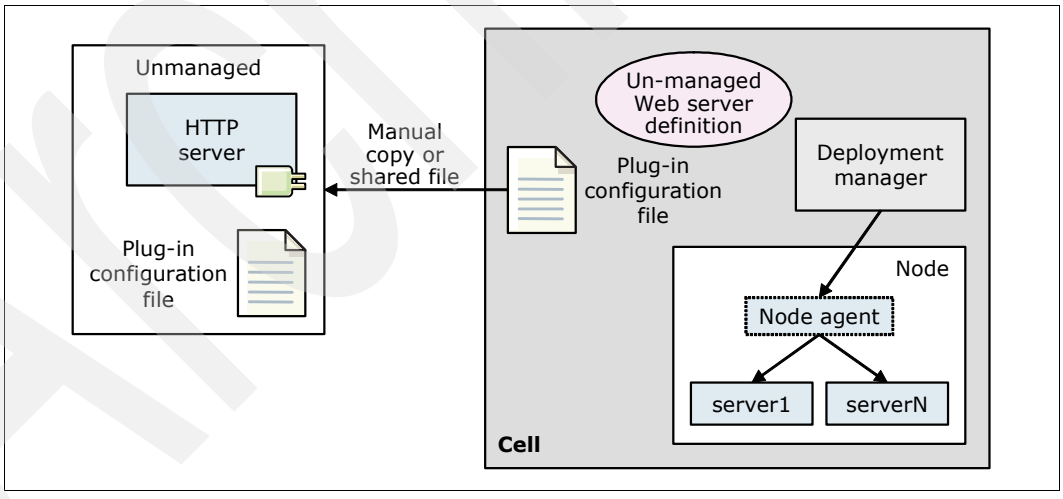

Figure 11 Unmanaged web server on an unmanaged node

Web server plug-ins

A web server can be used to serve static contents and requests, such as HTML pages. When a request requires dynamic content, such as JSP or servlet processing, it must be forwarded to the WebSphere Application Server for handling.

To forward a request, a web server plug-in is generated by the WebSphere Application Server packages and then propagated to a web server. You propagate (manually or automatically with the deployment manager) an Extensible Markup Language (XML) configuration file

(configured on the WebSphere Application Server) to the web server plug-in directory. The plug-in uses the configuration file to determine whether a request is handled by the web server or an application server. When WebSphere Application Server receives a request for an application server, it forwards the request to the appropriate web container in the application server. The plug-in can use HTTP or HTTPS to transmit the request. See Figure 11 on page 18. The plug-in is also used for routing requests to one of multiple application servers, see Figure 12.

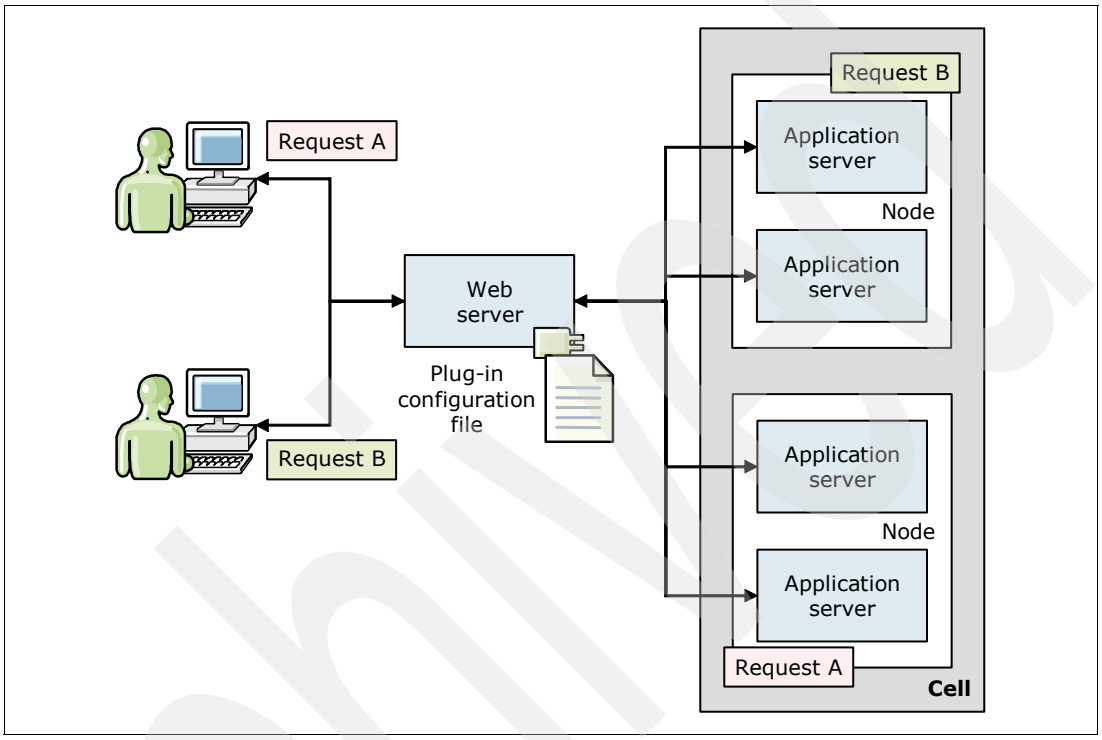

Figure 12 Web server plug-in concept with WebSphere Application Server

Generic servers

A *generic server* is a server that is managed in the WebSphere Application Server administrative domain even though the server is not supplied by WebSphere Application Server. The WebSphere Application Server generic servers function enables you to define a generic server as an application server instance within the WebSphere Application Server administration and associate it with a WebSphere Application Server or process.

There are two basic types of generic application servers:

- Non-Java applications or processes
- Java applications or processes

A generic server can be any server or process that is necessary to support the application server environment:

- Java server
- C or C++ server or process
- CORBA server
- Remote Method Invocation (RMI) server

Containers

Containers provide runtime support for applications. Runtime provisioning can be used to avoid starting the container if it is not needed.

In this section, we describe the container support for WebSphere Application Server V8.

Application server containers

Each application server provides the following container support:

- Web container

  The web container processes servlets, JSPs (processed as servlets), and other types of server-side includes. Each application server run time has one logical web container, which can be modified but not created or removed.

  Requests are received by the web container through the web container inbound transport chain. The chain consists of a TCP inbound channel that provides the connection to the network, an HTTP inbound channel that serves HTTP 1.0 and 1.1 requests, and a web container channel over which requests for servlets and JSPs are sent to the web container for processing. Requests for HTML and other static content directed to the web container are served by the web container inbound chain.

  It is suggested that you use an external web server to receive client requests and a web server plug-in to forward requests for servlets to the web container.

- Enterprise JavaBeans (EJB) container

  The EJB container provides all of the runtime services that are needed to deploy and manage enterprise beans. It is a server process that handles requests for both session and entity beans.

  The container provides many low-level services, including transaction support. From an administrative viewpoint, the container manages data storage and retrieval for the contained enterprise beans. A single container can host more than one EJB Java archive (JAR) file.

- Portlet container

  The portlet container provides the runtime environment to process JSR 286-compliant portlets. The portlet container is an extension to the web container. A simple portal framework is built on top of the container to render a single portlet into full browser page.

- SIP container

  The SIP container processes applications that use at least one SIP servlet written to the JSR 116 specification. It provides network services over which it receives requests and sends responses. It determines which applications to invoke and in what order. The container supports UDP, TCP, and TLS/TCP protocols.

- Batch container

  The batch container provides an execution environment for the execution of Java EE based batch applications. Batch applications are deployed as EAR files and follow either the transactional batch or compute-intensive programming models.

Workload management

Workload management (WLM) is the concept of sharing requests across multiple instances of a resource. Workload management techniques are implemented expressly for providing scalability and availability within a system. These techniques allow the system to serve more concurrent requests.

WLM provides the following main features:

- *Load balancing* is the ability to send the requests to alternate instances of a resource. Workload management allows for better use of resources by distributing loads more evenly. Components that are overloaded, and therefore a potential bottleneck, can be routed around with workload management algorithms. Workload management techniques also provide higher resiliency by routing requests around failed components to duplicate copies of that resource.
- *Affinity* is the ability to route concurrent requests to the same component that has served the first request. A web client can establish session affinity for requests to be routed to a particular application server.

HTTP servers

An IP sprayer component, such as the Edge Components Load Balancer or a network appliance, can perform load balancing and workload management functionality for incoming web traffic, as shown on Figure 13.

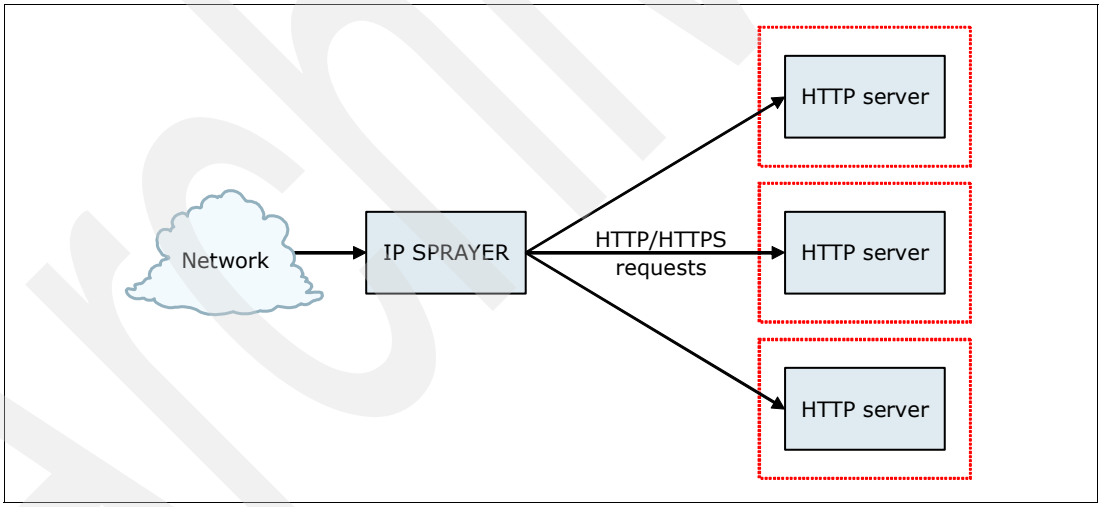

Figure 13 IP Sprayer/HTTP server workload management

Depending on which solution you implement, you can have the following routing options:

- *Dynamic weight*, where the load balancer calculates the load of each HTTP server and routes dynamically to the one that is less busy
- *Static weight*, where each member has a weight and the load balancer spreads the requests depending on this weight

These routing options are also impacted by the following affinity rules:

- Stickiness to source IP address
- Cookie affinity

- URI affinity
- SSL session ID

By default, affinity options overwrite any load-balancing options.

Clustering application servers

Clustering application servers that host web containers automatically enable plug-in workload management for the application servers and the servlets that they host. Routing of servlet requests occurs between the web server plug-in and the clustered application servers using HTTP or HTTPS, as shown in Figure 14.

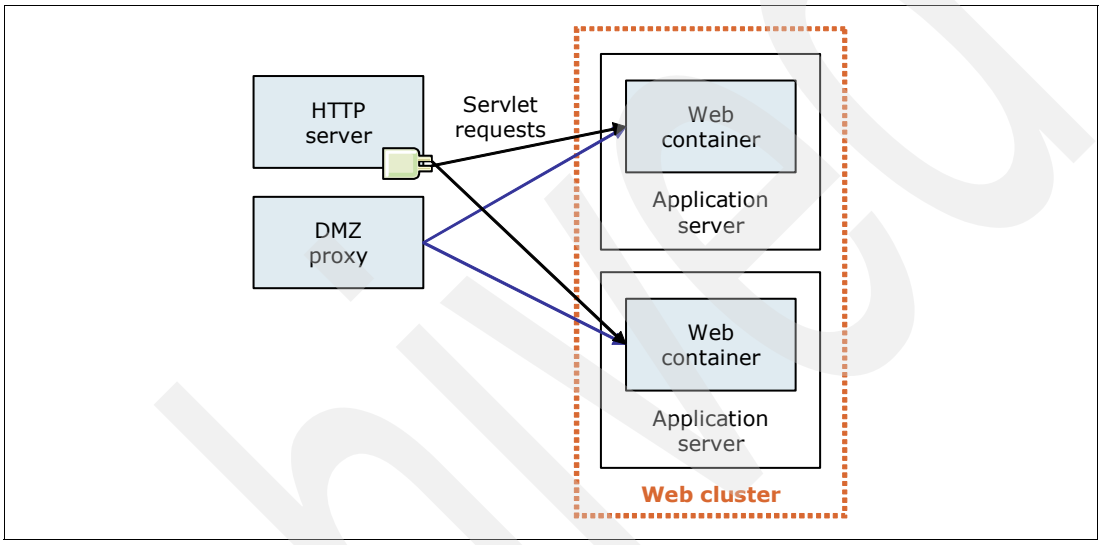

Figure 14 Plug-in (web container) workload management

WebSphere provides two load balancing options:

- Round-robin

  This routing is based on the weight that is associated with the cluster members. If all cluster members have identical weights, the plug-in sends equal requests to all members of the cluster, assuming no strong affinity configurations. If the weights are scaled in the range from 0 to 20, the plug-in routes requests to those cluster members with the higher weight value more often. No requests are sent to cluster members with a weight of 0 unless no other servers are available. Round-robin is the default load balancing policy.

  A guideline formula for determining routing preference is:

  `% routed to Server1 = weight1 / (weight1+weight2+...+weight`*n*`)`

  Where there are *n* cluster members in the cluster.

- Random

  The plug-in picks a member of the cluster randomly.

The load balancing options are impacted by session affinity. After a session is created with an initial request, all subsequent requests are served by the same cluster member. The plug-in identifies the application server that served the previous request by analyzing the session identifier and attempts to route all subsequent requests to this server.

Workload management for EJB containers can be performed by configuring the web container and EJB containers on separate application servers. Multiple application servers

with the EJB containers can be clustered, enabling the distribution of EJB requests between the EJB containers, shown in Figure 15. Be aware there might be a performance degradation because the EJB calls are made out of the process.

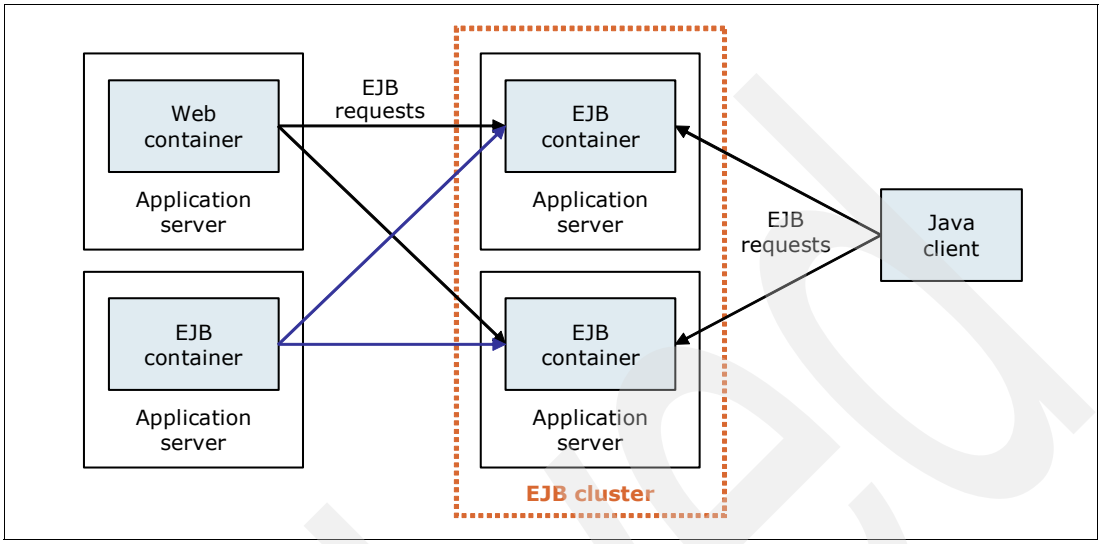

Figure 15 EJB workload management

To route EJB requests, WebSphere Application Server provides the following main routing policies:

- Server weighted round-robin routing

  In this configuration, EJB client requests are routed to available EJB servers in a round-robin fashion based on assigned server weights. The EJB clients can be servlets operating within a web container, stand-alone Java programs using RMI/IIOP, or other EJBs.

  The server weighted round-robin routing policy ensures a distribution based on the set of server weights that are assigned to the members of a cluster. For example, if all servers in the cluster have the same weight, the expected distribution for the cluster is that all servers receive the same number of requests. If the weights for the servers are not equal, the distribution mechanism sends more requests to the higher weight value servers than the lower weight value servers. This policy ensures the desired distribution based on the weights assigned to the cluster members.

- Prefer local routing

  You can also choose that EJB requests are preferably routed to the same host as the host of the requesting EJB client. In this case, only cluster members on that host are chosen (using the round-robin weight method). Cluster members on remote host are chosen only if a local server is not available.

The following affinity policies impact the routing:

- Process affinity: If an EJB is available in the same cluster member as the client, all requests from that client are directed to the EJB in the same JVM process. One of the advantages of the policy is there is no need for serialization of method calls.
- Transaction affinity: All the requests from the same transaction are directed to the same cluster member. This policy overrides all the other policies.

High availability

High availability (HA) is also known as *resiliency*. HA is the description of the system's ability to tolerate a certain amount of failures and remain operational and is achieved by adding redundancy in the infrastructure to support failures. Availability impacts both performance and scalability. Depending on your needs, you must define the level of HA of your infrastructure. Keep in mind that the overall infrastructure is available only if all the components are available.

WebSphere Application Server uses a high availability manager (HA manager) to eliminate single points of failure. The HA manager is responsible for running key services on available application servers rather than on a dedicated server (such as the deployment manager). It continually polls all of the core group members to verify that they are active and healthy. The HA manager service runs by default in each server, as shown in Figure 16. A core group is an HA domain that consists of a set of processes in the same cell that can directly establish HA relationships.

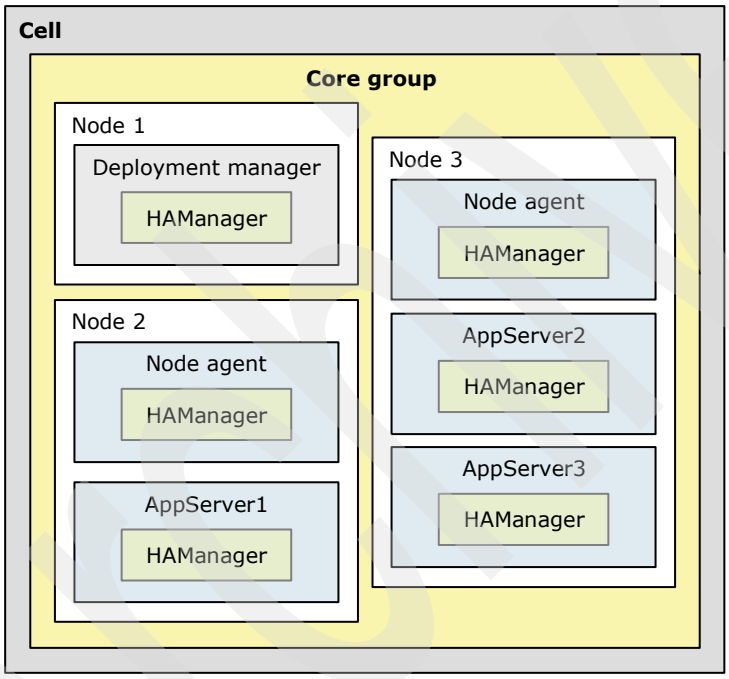

Figure 16 Core group concept

For certain functions (such as transaction peer recovery) the HA manager takes advantage of fault tolerant storage technologies, such as Network Attached Storage (NAS), which can significantly lower the cost and complexity of high availability configurations. The HA manager also provides peer-to-peer failover for critical services by maintaining a backup for these services. WebSphere Application Server also supports HA solutions such as Power HA, IBM Parallel Sysplex®, and others.

An HA manager continually monitors the application server environment. If an application server component fails, the HA manager takes over the in-flight and in-doubt work for the failed server. This process introduces some overhead but significantly improves application server availability.

An HA manager focuses on recovery support and scalability in the following areas:

- Embedded messaging
- Transaction managers
- Workload management controllers
- Application servers
- WebSphere partitioning facility instances
- On-demand routing
- Memory-to-memory replication through data replication service
- Resource adapter management

2-phase commit process

WebSphere Application Server V8.0 is a transaction manager that supports the coordination of resource managers through their XAResource interface and participates in distributed global transactions with transaction managers that support the CORBA Object Transaction Service (OTS) protocol or Web Service Atomic Transaction (WS-AtomicTransaction) protocol. WebSphere Application Server also participates in transactions that are imported through Java EE Connector 1.5 resource adapters. Resource managers support two-phase coordination by offering an XAResource interface. WebSphere Application Server transaction support provides coordination, within a transaction, for any number of two-phase capable resource managers.

Administration

WebSphere Application Server V8.0 provides a variety of administrative tools for configuring and managing your runtime environment:

- Integrated Solutions Console

  The Integrated Solutions Console is a browser-based client that uses a web application running in the web container to administer WebSphere Application Server.

- WebSphere scripting client (**wsadmin**)

  The **wsadmin** client is a non-graphical scripting and interactive interface that can be used to administer WebSphere Application Server from a command line prompt. It can connect to WebSphere Application Server using one of the two communication mechanisms:

  – Using SOAP (the default) by communicating with the embedded HTTP server in the web container.

  – Using Remote Method Invocation (RMI) to communicate with the administrative services.

  Two scripting languages are supported:

  – Jython
  – JACL

- Task automation with Ant

  With Another Neat Tool (Ant), you can create build scripts that compile, package, install, and test your application on WebSphere Application Server.

- Administrative applications

  You can develop custom Java applications that use the Java Management Extensions (JMX) based on the WebSphere application programming interface (API).

- Command-line utilities

  WebSphere Application Server provides administrative utilities to help manage your environment. They offer the following features:

  - Called from a command line
  - Performs common administrative tasks, such as starting and stopping WebSphere Application Server, backing up the configuration, and other tasks
  - Works on local servers and nodes only, including the deployment manager

- Centralized installation manager

  The centralized installation manager (CIM) is used to consolidate and simplify the steps that are required to perform installations and to apply maintenance on systems.

- Job Manager

  The job manager provides a centralized interface for asynchronous job submissions. It can submit CIM jobs to multiple cells and server topologies.

- WebSphere Customization Toolbox

  The WebSphere Customization Toolbox includes tools for customizing various parts of the WebSphere Application Server environment. You can launch the Web Server Plug-ins Configuration Tool, Profile Management Tool (z/OS only), and the z/OS Migration Management Tool.

- Property file based management

  Properties files can be used to manage environment and configuration objects. Configuration objects can be extracted to and modified in simple properties file format and can be applied to update the system configuration.

The choice of which combination of administrative tools you employ depends on the size and complexity of your runtime environment.

There are multiple levels of administration in WebSphere Application Server:

- In the WebSphere Application Server Express and Base packages, you can administer stand-alone server instances individually.
- In the WebSphere Application Server Express and Base packages, you can administer multiple stand-alone server instances on a single system using an administrative agent.
- In the WebSphere Application Server Network Deployment package, you can administer an entire cell of application servers using a deployment manager.
- You can administer multiple stand-alone application servers, administrative agents, and deployment managers using a job manager.

Web services

Web services are self-contained, modular applications that can be described, published, located, and invoked over a network. WebSphere Application Server supports SOAP-based web service hosting and invocation.

WebSphere Application Server can act as both a web service provider and as a requester. As a requester, WebSphere Application Server hosts applications that invoke web services from other locations. As a provider, WebSphere Application Server hosts web services that are published for use by clients.

WebSphere Application Server V8 supports the Web Services for Java Enterprise Edition (Java EE) V1.3 specification. This specification defines the programming model and runtime architecture to deploy and look up web services in the Java EE (JEE) environment. More specifically, to deploy and look up web services in the JEE environment on the web, EJB, and client application containers.

Web services support in WebSphere Application Server V7.0 and V8.0 also includes the following standards and specifications:

- Java API for XML Web Services (JAX-WS)

  The core programming model and bindings for developing and deploying web services on the Java platform.

- Java API for RESTful Web Services (JAX-RS)

  The programming model for developing and deploying web services based on the Representation State Transfer (REST) architectural style.

- WS Transaction support

  Defines how web services applications can work within global transactions in enterprise environments using the following three specifications:

  - WS-Atomic Transaction (WS-AT)

    A specific coordination type that defines protocols for atomic transactions.

  - WS-Business Activity (WS-BA)

    A specific coordination type that defines protocols for business activities. A business activity is a group of general tasks that you want to link together so that the tasks have an agreed outcome.

  - WS-Coordination (WS-Coor)

    Specifies a context and a registration service with which participant web services can enlist to take part in the protocols that are offered by specific coordination types.

- WS-I Basic Profile

  A set of non-proprietary web services specifications that promote interoperability.

- WS- Notification

  Publish and subscribe messaging for web services.

- WS-Addressing

  Enables systems to support message transmission and identification through networks that include firewalls or gateways in a transport-neutral manner.

- WS-Security

  This specification covers a standard set of SOAP extensions that can be used when building secure web services to provide integrity and confidentiality. It is designed to be open to other security models including PKI, Kerberos, and SSL. WS-Security provides support for multiple security tokens, multiple signature formats, multiple trust domains, and multiple encryption technologies. It includes security token propagation, message integrity, and message confidentiality.

- WS-Policy

  This standard is used to describe and communicate web service policies. It allows service providers to export policy requirements in a standard format. Clients can combine their capabilities with service provider requirements to establish policies required for a a specific interaction.

For a complete list of supported standards and specifications, see the web services section at:

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/rovr_specs.html

Messaging

WebSphere Application Server applications invoke asynchronous messaging services by using the Java Messaging Service application program interface (JMS API) to interface with a messaging provider. There are three types of messaging providers:

- Default messaging provider
- WebSphere MQ messaging provider
- Third-party messaging provider

Figure 17 show an example of an application using a default messaging provider as a messaging provider.

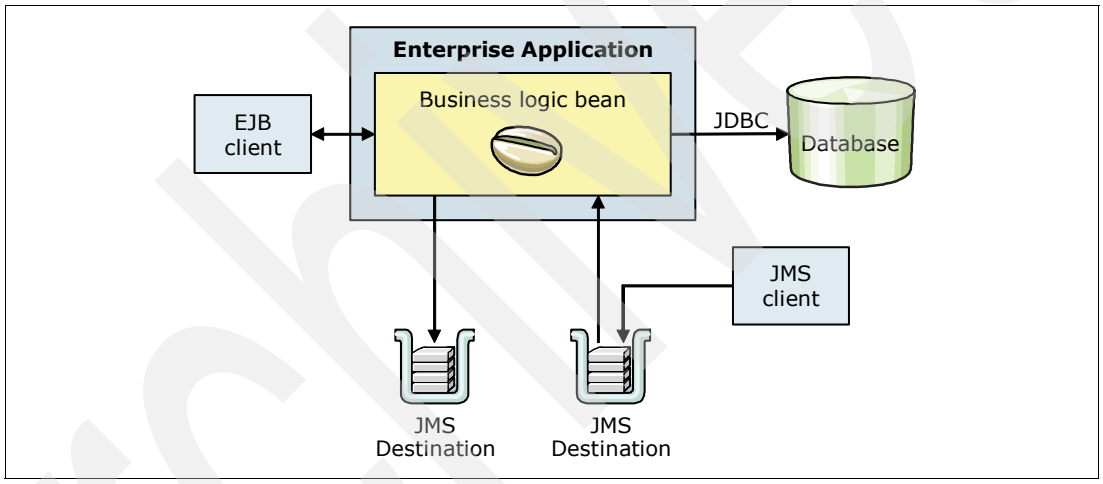

Figure 17 Default messaging provider

Applications can use the following styles of asynchronous messaging:

- Point-to-point

  Point-to-point applications typically use queues to pass messages to each other. An application sends a message to a destination queue. The receiving application can then retrieve the message from the queue

- Publish/subscribe

  In publish/subscribe messaging, there are two types of applications: publisher and subscriber. A publisher supplies information in the form of messages. A subscriber is a customer of the information that is published.

Service integration

Service integration technology provides the communication infrastructure for messaging and service-oriented applications, unifying this support into a common component. Service integration includes the following features:

- A JMS 1.1 compliant JMS provider

  This provider is referred to as the *default messaging provider*.

- The service integration bus (referred to as the *bus*)

  The service integration bus provides the communication infrastructure for the default messaging provider. The bus supports the attachment of web services requestors and providers.

- Support for the web services gateway

  This provides a single point of control, access, and validation of web service requests. It enables you to control which web services are available to groups of web service users.

Service integration bus

Service integration bus capabilities are fully integrated into WebSphere Application Server, enabling it to take advantage of WebSphere security, administration, performance monitoring, trace capabilities, and problem determination tools. Figure 18 illustrates the service integration bus and how it fits into the larger picture of an enterprise service bus.

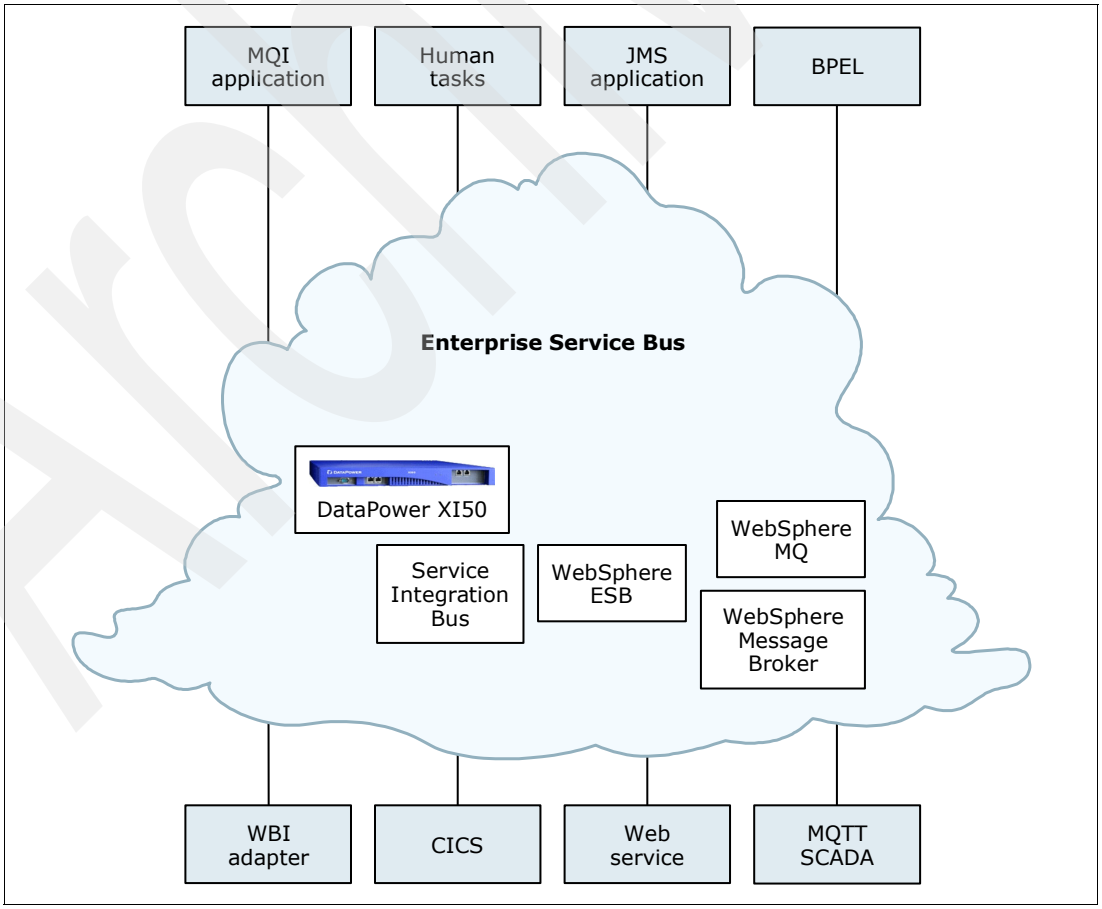

Figure 18 The enterprise service bus

A service integration bus consists of the following components:

- Bus members

  Application servers or clusters that were added to the bus.

- Foreign bus

  A foreign bus is an external messaging product that is either another bus or a WebSphere MQ network.

- Messaging engine

  The application server or cluster component that manages bus resources. When a bus member is defined, a messaging engine is created automatically on the application server or cluster. The messaging engine provides a connection point for clients to either produce or to consume messages.

  An application server has one messaging engine per bus of which it is a member. A cluster has at least one messaging engine per bus and can have more.

- Destinations

  A destination is the place within the bus to which applications attach to exchange messages. Destinations can represent web service endpoints, messaging point-to-point queues, or messaging publish/subscribe topics. Destinations are created on a bus and hosted on a messaging engine.

- Message store

  A messaging engine uses a message store for message persistence and to save information that is needed for recovery in the event of a failure (including messages, subscription information, and transaction states). Each messaging engine has only one message store, which can be either file-based or a database.

  With a file store (the default), information is stored in a file system through the operating system.

  With a database-based message store, information is stored in tables of a relational database. Multiple messaging engines can share a database for the data store, each with having its own set of tables and schema.

The service integration bus supports the following application attachments:

- Messaging applications

  JMS applications running in WebSphere Application Server can connect to the bus using the JMS programming model.

- Web services:
  - Requestors using the JAX-RPC API
  - Providers running in WebSphere Application Server as stateless session beans and servlets (JSR-109)
  - Requestors or providers attaching through SOAP/HTTP or SOAP/JMS

Service integration bus and messaging

With the Express or Base packages, you typically have one stand-alone server with one messaging engine on one service integration bus. With the Network Deployment package, you have more flexibility to use multiple buses for high availability and scalability.

Figure 19 illustrates two application servers, each with a messaging engine on a service integration bus.

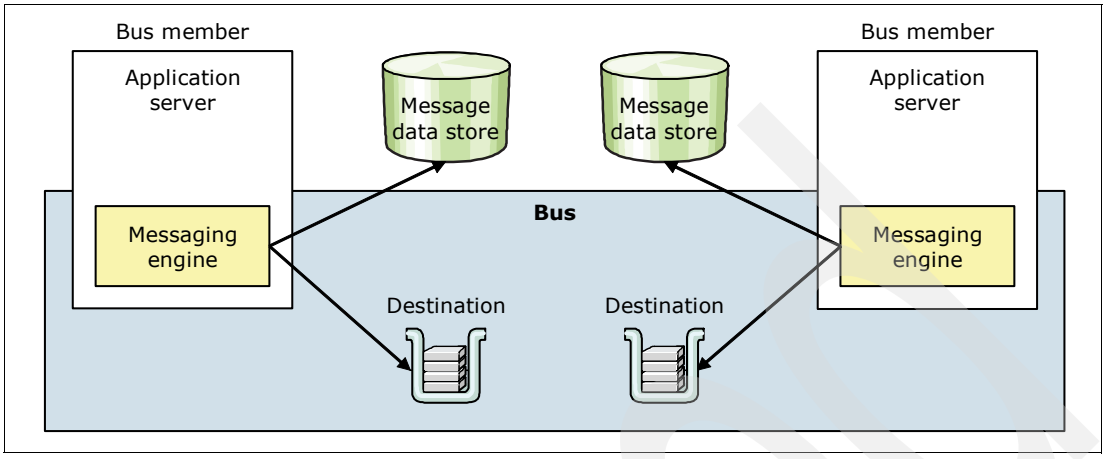

Figure 19 Service integration bus

The following topologies are valid:

- One bus and one messaging engine (application server or cluster)
- One bus with multiple messaging engines
- Multiple buses within a cell that might or might not be connected to each other
- Buses connected between cells
- One application server that is a member of multiple buses and that has one messaging engine per bus
- A connection between a bus and a WebSphere MQ queue manager

  When using this type of topology, consider the following points:

  – A messaging engine cannot participate in a WebSphere MQ cluster.
  – You can configure the messaging engine to look like another queue manager to WebSphere MQ.
  – WebSphere applications can send messages directly to WebSphere MQ or though the service integration bus.
  – You can use a JMS thin client as opposed to a full WebSphere Application Server client installation.

For a more complete discussion on the differences between the service integration bus and WebSphere MQ, refer to the following usage paper:

http://www-01.ibm.com/support/docview.wss?uid=swg21497341

Clustering

In a distributed server environment, you can use clustering for high availability and scalability. You can add a cluster as a bus member and achieve the following results:

- High availability

  One messaging engine is active in the cluster. In the event that the messaging engine or server fails, the messaging engine on a standby server is activated.

- Scalability

  A single messaging destination can be partitioned across multiple active messaging engines in the cluster. Each message engine cannot failover to any other server in the cluster. Messaging order is not preserved.

- Scalability and high availability

  One message engine is created for each application server in the cluster. Each message engine can failover to any other server in the cluster. Messaging order is not preserved.

Quality of service

You can define quality of service on a destination basis to determine how messages are (or are not) persisted. You can also specify quality of service within the application.

Message driven beans

Message driven beans (MDB) in the application server that listen to queues and topics are linked to the appropriate destinations on the service integration bus using JCA connectors (ActivationSpec objects).

Security

WebSphere Application Server Security Infrastructure provides several services. The services are integrated with the underlying Operating System, the runtime environment, and with other servers and server components. The key security areas are the following:

- Authentication
- Authorization
- Key and certificate management
- Auditing
- Flexible configuration option

Authentication

Authentication is the process of confirming a user or system identity. The authentication mechanism in WebSphere Application Server uses a user registry to perform this validation. A successful authentication results in the creation of a credential, which is the internal representation of a successfully authenticated client user. The abilities of the credential are determined by the configured authorization mechanism.

The WebSphere application server supports three types of web login authentication mechanisms:

- Basic Authentication
- Certificate-based Authentication
- Form-based Authentication

WebSphere Application Server V8 supports a number of authentication mechanisms, but not all of them are directly selected in the administrative console:

- Lightweight Third Party Authentication (LTPA)
- Kerberos
- Simple and Protected GSSAPI Negotiation Mechanism (SPNEGO)
- Rivest Shamir Adleman (RSA) token authentication
- Web Services Security SAML Token Profile

Authorization

Authorization is the process of checking whether a given user has the privileges necessary to access a requested resource. In the case of WebSphere Application Server, we differentiate between two kinds of authorization, according to two types of users:

- Administrative user authorization

  The administrative security in WebSphere Application Server controls access to the configuration and management interfaces.

  Fine-grained administrative security can grant access to each user role per resource instance, instead of granting access to all of the resources in the cell. This allows for a better separation of administrative duties.

- Application user authorization

  The Java EE specification defines the building blocks and elements of a Java EE application. The specification provides the details about security that is related to several elements. The application maps user roles with the resource access rights, in development time. During application deployment the administrator or deployer maps the user roles to real users and groups.

Key and certificate management

SSL is the industry standard for data interchange encryption between clients and servers. In WebSphere Application Server V8, Java Secure Sockets Extension (JSSE) is used to handle the handshake negotiation and protection capabilities that are provided by SSL to ensure that secure connectivity exists across most protocols. SSL security can be used for establishing communications inbound to and outbound from an endpoint. To establish secure communications, a certificate and an SSL configuration must be specified for the endpoint.

WebSphere Application Server differentiates some kinds of communication:

- Internal management communication

  The WebSphere Application Server uses SSL to communicate within the cell among the nodes. It maintains certificates for each node in the cell.

  When a new profile is created, including the deployment manager profile, a new unique chained certificate is also generated for the profile. This certificate consists of a signer certificate, which has a 15 year expiration time and personal server certificates that have a one year expiration time, by default. WebSphere Application Server has its own, built-in mini CA with which it signs the certificates in the cell.

  Alternatively the customer can use their own certificate settings. In this case, the customer can import their own certificates and override some settings.

- External service communication

  These components are some of the external connections that require SSL encryption:

  - JDBC database connection
  - LDAP directory protocol connection
  - Messages channels
  - Web Services communication

Certificates can be created and managed through the Integrated Solutions Console. WebSphere Application Server provides mechanisms for creating and managing client CA clients and keystores and for creating self-signed certificates and certificate authority requests. Keystores in WebSphere Application Server profiles hold personal certificates,

while the trust stores store signer certificates from other servers with which it is communicating.

All certificates have expiration time. The Server Personal Certificate has a default of one year and the signer certification has a default of 15 years. Certificate replacement can be done manually, but the more effective method is to let the application server replace the certificates automatically, which is done with the help of the built in expiration manager.

Auditing

WebSphere Application Server auditing works through event logging. All security related events are filtered with an audit filter and an event outcome filter. Captured events, which go through both filters, are logged into the audit log. The log file can be encrypted to avoid unauthorized read access and also can be signed to block unauthorized write access.

WebSphere Application Server V8 has a built-in *auditor* administrative role. Only the administrators in the auditor role can change the audit subsystem related settings and review the audit logs.

Flexible configuration option

One exciting improvement in WebSphere Application Server V8 is the capability to use security domains. Each security domain can have its own, separately configured Virtual Member Manager (VMM) instance.

WebSphere Application Server V8 has several options and combinations to select the best user registry setting for an application environment.

Security domains

The WebSphere security domains (WSD) provide the flexibility to use several security configurations in a WebSphere Application Server cell. WSD is also referred to as multiple security domains, or simply, security domains. With security domains you can configure several security attributes, such as the user registry, for various applications in the same cell.

The global security configuration applies to all administrative functions, naming resources, and Mbeans, and is the default security configuration for user applications. One global security configuration must be defined before the security domains can be created. If no security domains are configured, all of the applications use information from the global security configuration. When a security domain is created and associated with a scope, only the user applications in that scope use the security attributes that are defined in the security domain. The administrative applications and the naming operations in that scope use the global security configuration. Each security domain must be associated with a scope (cell, or specific clusters, servers, and service integration buses) where it will be applied.

User registries

The information about users and groups reside in a user registry. In WebSphere Application Server, a user registry authenticates a user. It contains information about users and groups so that security-related functions, including authentication and authorization can be performed.

Although WebSphere Application Server supports several types of user registries, only one can be active in a certain scope. WebSphere Application Server supports the following types of user registries:

- Local operating system
- Standalone Lightweight Directory Access Protocol (LDAP)
- Federated repository (a combination of a file-based registry and one or more LDAP servers in a single realm)
- Custom registry

Application development and deployment

There are several tools in the WebSphere Application Server V8.0 environment that help in the development and deployment of applications. All editions of WebSphere Application Server V8.0 include a fully licensed version of IBM Assembly and Deploy Tools for WebSphere Administration and a 60-day trial version of Rational Application Developer Standard Edition for WebSphere Software V8:

- IBM Assembly and Deploy Tools for WebSphere Administration

  IBM Assembly and Deploy Tools for WebSphere Administration is targeted for the assembly and deployment of applications only. It does not provide development capabilities. The key components of this tool are:

  - Import and validate applications
  - Edit deployment descriptors and binding files
  - Edit EAR-level configuration (Enhanced EAR)
  - Create and debug Jython and wsadmin scripts
  - Deploy EJB and web services
  - Deploy applications to local or remote WebSphere Application Server V8.0 servers
  - Debug applications on WebSphere Application Server V8.0

- Rational Application Developer Standard Edition for WebSphere Software V8.0

  Rational Application Developer for WebSphere Software Standard Edition V8.0 includes all of the features of IBM Assembly and Deploy Tools for WebSphere Administration. It also includes the capabilities to design and develop applications. Additionally, it provides productivity enhancement features to quickly begin application development and testing. It includes full support for Java EE 6, including EJB 3.1 and Java Development Kit (JDK) 6.

- Rational Application Developer for WebSphere Software V8.0

  Rational Application Developer for WebSphere Software V8.0 offers all of the features that are included in Rational Application Developer Standard Edition plus additional capabilities for team productivity, problem determination, and enterprise connectivity, such as WebSphere Adapter Support and integration with IBM Rational Team Concert™ and IBM Rational ClearCase®.

Source code management

Support for team development is provided by source code management (SCM) systems. Rational Application Developer for WebSphere Software V8 supports the following SCM systems:

- Rational TeamConcert

  Rational Team Concert is built on the IBM Jazz™ platform, which provides a common collaboration environment to improve communication across the teams in an organization. With efficient communication during the development of your applications, it is possible to produce quality software that satisfies all requirements and stakeholders expectations

more easily. It also has its own SCM system that can help support geographically distributed teams.

- Rational ClearCase

  Rational ClearCase organizes its code repositories as Versioned Object Bases (VOBs). VOBs contain versioned file and directory elements. Users of Rational ClearCase are organized according to their roles. Each user has their own view of the data that is in the VOB on which they are working. Rational ClearCase tracks VOBs and views. It also coordinates the checking in and checking out of VOB data to and from views.

- Concurrent Versions System (CVS)

  CVS uses a branch model to support multiple courses of work that are somewhat isolated from each other but still highly interdependent. Branches indicate where a development team shares and integrates ongoing work. A branch can be thought of as a shared workspace updated by team members as they make changes to the project. This model enables individuals to work on a CVS team project, share their work with others as changes are made, and access the work of others as the project evolves.

- Subversion

  Subversion is an open source version control system that tracks the entire file system and files. It versions directories and individual files and stores them into a repository.

Application deployment

Applications are installed on application servers using the administrative console or the **wsadmin** scripting interface. You can deploy an application to a single server or a cluster. In the case of a cluster, an application is installed on each application server in the cluster. Installing an application involves the following tasks:

- Binding resource references (created during packaging) to actual resources (for example, a data source must be bound to a real database)
- Defining JNDI names for EJB home objects
- Specifying data source entries for entity beans
- Binding EJB references to the actual EJB JNDI names
- Mapping web modules to virtual hosts
- Specifying listener ports for message-driven beans
- Mapping application modules to application servers
- Mapping security roles to users or groups

After you deploy a new application, you must regenerate and propagate the web server plug-in configuration file to the web server.

Monitored directory support

Using a monitored directory, WebSphere Application Server can simplify the process of editing, compiling, deploying, debugging, updating, and uninstalling applications. When an application is moved into a monitored directory and after a defined polling interval, it is installed and started automatically. Likewise, if the application is removed from the directory, it is stopped and uninstalled. If the application or module is moved into the directory again, it is updated.

Monitored directories support the following file types:

- Enterprise archive (EAR)

- Web archive (WAR)
- Java archive (JAR)
- SIP Application Resource (SAR)

Application update

WebSphere Application Server allows partial updates to applications and makes it possible to restart only parts of an application. Updates to an application can consist of individual application files, application modules, compressed files that contain application artifacts, or the complete application. All module types can be started (but only web modules can be stopped).

WebSphere Application Server has a rollout start option for installing applications on a cluster that can stop, update, and start each cluster member in turn, ensuring availability.

WebSphere Application Server feature packs

A WebSphere Application Server feature pack is an optionally installable product extension for WebSphere Application Server that provides a set of new related standards and innovative features. With feature packs, users can take advantage of these standards and features without waiting for the next release of WebSphere Application Server.

The WebSphere Application Server Feature Pack for Web 2.0 and Mobile version 1.1.0 is currently available for WebSphere Application Server V8.0. This feature pack extends SOA by connecting external web services, internal SOA services, and Java EE objects into highly-interactive web application interfaces. It provides a supported, best-in-class Ajax development toolkit for WebSphere Application Server and also a rich set of extensions to Ajax.

Figure 20 shows the main components of this feature pack.

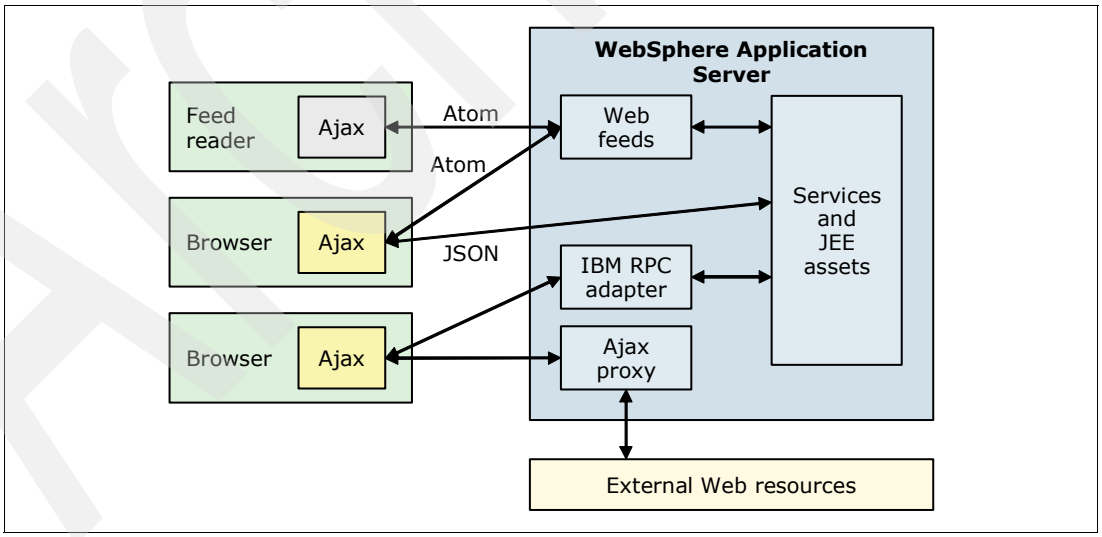

Figure 20 Components of WebSphere Application Server Feature Pack for Web 2.0 and Mobile

The following previously available feature packs for WebSphere Application Server V7.0 are integrated into WebSphere Application Server V8.0; therefore, you can use their features without additional installation procedures:

- WebSphere Application Server Feature Pack for Communications Enabled Applications

- WebSphere Application Server Feature Pack for Modern Batch
- WebSphere Application Server Feature Pack for OSGi Applications and JPA 2.0
- WebSphere Application Server Feature Pack for SCA
- WebSphere Application Server Feature Pack for XML
- WebSphere Application Server Feature Pack for Dynamic Scripting

Integrating with other products

WebSphere Application Server works closely with other IBM products to provide a fully integrated solution. This section introduces some of these products, including those that provide enhanced security, messaging options, and broad integration features.

IBM Tivoli Access Manager

IBM Tivoli® Access Manager provides centralized authentication and authorization services.

The WebSphere Application Server security infrastructure is adequate for many situations and circumstances. However, integrating WebSphere Application Server with Tivoli Access Manager allows for an end-to-end integration of application security across the entire enterprise.

Tivoli Access Manager uses the following components:

- User repository

  Tivoli Access Manager requires a user repository, such as IBM Tivoli Directory Server or Microsoft Active Directory. Tivoli Access Manager can be configured to use the same user repository as WebSphere Application Server, enabling you to share user identities with both Tivoli Access Manager and WebSphere Application Server.

- Tivoli Access Manager policy server

  This component maintains the master authorization policy database, which contains the security policy information for all resources and all credentials information of all participants in the secure domain (both users and servers).

- Tivoli Access Manager client

  This client is embedded in WebSphere Application Server. The Tivoli Access Manager client can be configured using the scripting and GUI management facilities of WebSphere Application Server.

Figure 21 on page 39 shows the integration interfaces between WebSphere Application Server and Tivoli Access Manager.

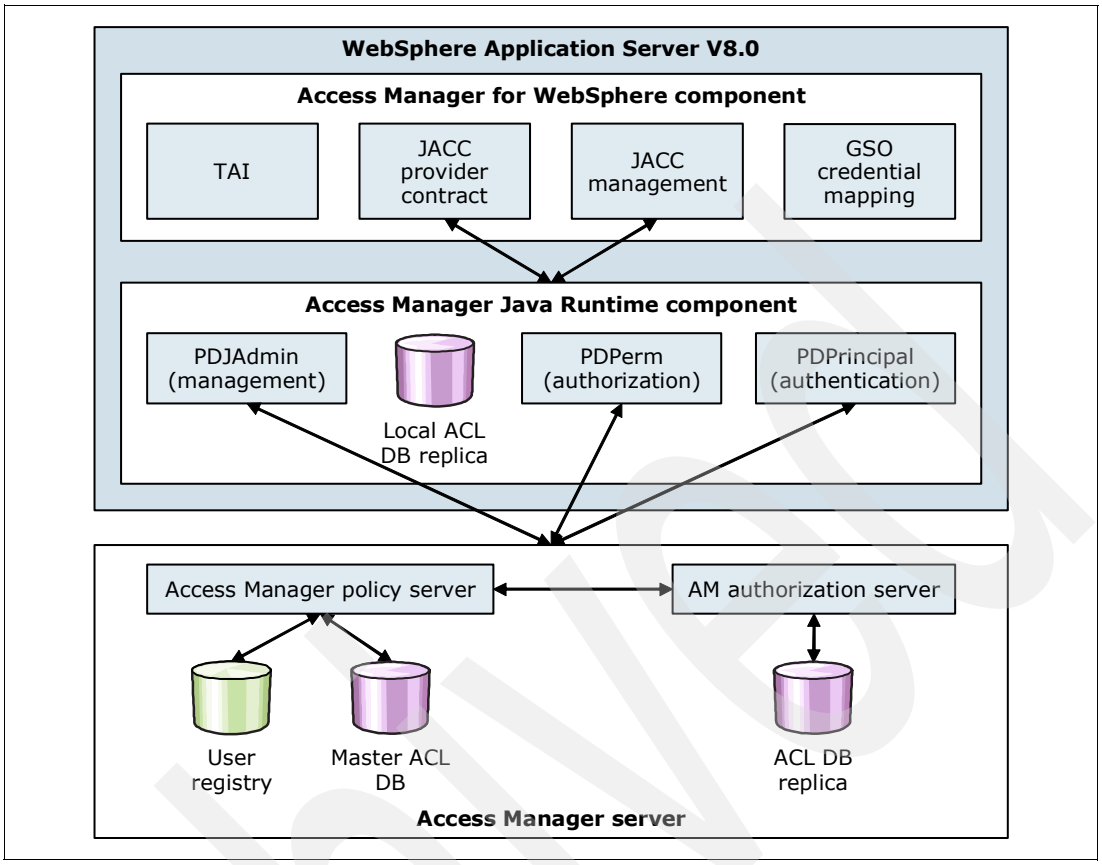

Figure 21 Integration of WebSphere Application Server with Tivoli Access Manager

IBM Tivoli Directory Server

IBM Tivoli Directory Server provides a high-performance LDAP identity infrastructure that can handle millions of entries. It is built to serve as the identity data foundation for web applications and identity management initiatives.

IBM WebSphere MQ

IBM WebSphere MQ is an asynchronous messaging technology that is available from IBM. WebSphere MQ is a middleware technology designed for application-to-application communication rather than application-to-user and user interface communication.

WebSphere MQ is available on a large number of platforms and operating systems. It offers a fast, robust, and scalable messaging solution that assures once, and once only, delivery of messages to queue destinations that are hosted by queue managers. This messaging solution has APIs in C, Java, COBOL, and other languages, which allow applications to construct, send, and receive messages.

WebSphere MQ can be configured as a messaging provider in WebSphere Application Server (in addition to or as an alternative to the default messaging provider). If you are using the default messaging provider, there are also two mechanisms available to connect the default messaging provider with a WebSphere MQ network:

- Extend the WebSphere MQ and service integration bus networks by defining a WebSphere MQ link on a messaging engine in a WebSphere Application Server that connects the service integration bus to a WebSphere MQ queue manager.
- Integrate specific WebSphere MQ resources into a service integration bus for direct, synchronous access from default messaging applications running in WebSphere Application Servers. This is achieved by representing a queue manager or queue sharing group as a WebSphere MQ server in the WebSphere Application Server cell and adding it to a service integration bus as a bus member.

Figure 22 shows a sample integration for WebSphere Application Server and WebSphere MQ.

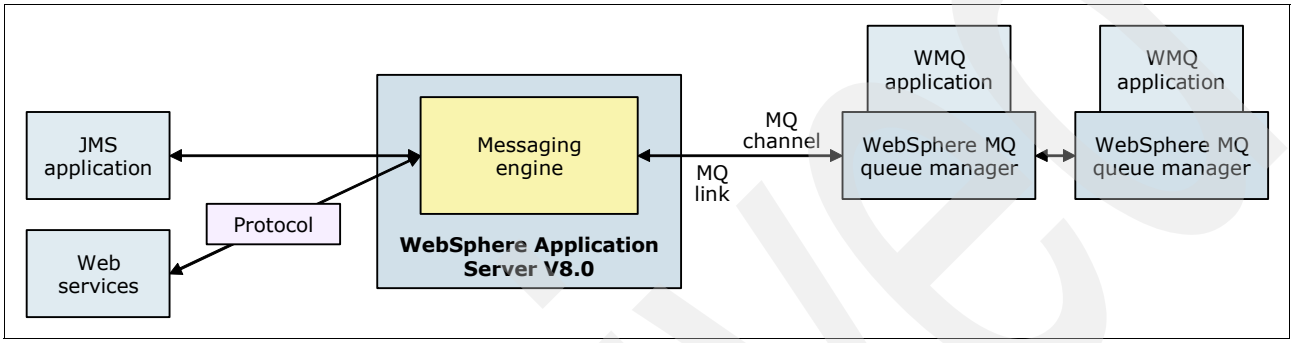

Figure 22 WebSphere Application Server integration with WebSphere MQ

IBM WebSphere Adapters

A resource adapter is a system-level software driver that a Java application uses to connect to an enterprise information system (EIS). A resource adapter plugs into an application client and provides connectivity between the EIS and the enterprise application. WebSphere Application Server includes a development license for IBM WebSphere Adapters.

IBM WebSphere Adapters provide a set of generic technology and business application adapters with wizards that quickly and easily service enable enterprise information systems (EISs), such as existing applications, enterprise resource planning (ERP), Human Resources (HR), Customer Relationship Management (CRM), and supply chain systems. WebSphere Adapters can also integrate those systems to IBM Business Process Management, WebSphere Enterprise Service Bus, and application server solutions in a service-oriented architecture (SOA).

IBM WebSphere DataPower

IBM WebSphere DataPower® SOA Appliances represent an important element in the IBM holistic approach to SOA. IBM SOA appliances are purpose-built, easy-to-deploy rack-mountable 1U network hardware devices or blade servers that simplify, secure, and accelerate your XML and web services deployments while extending your SOA infrastructure. These new appliances offer an innovative, pragmatic approach to harness the power of SOA while simultaneously enabling you to use your existing application, security, and networking infrastructure investments.

The Integrated Solutions Console contains an administration interface called the DataPower appliance manager, to manage multiple WebSphere DataPower appliances.

IBM Tivoli Composite Application Manager for WebSphere

IBM Tivoli Composite Application Manager for WebSphere (ITCAM for WebSphere) is an application management tool that helps maintain the availability and performance of on demand applications. It helps you pinpoint, in real time, the source of bottlenecks in application code, server resources, and external system dependencies. ITCAM for WebSphere provides in-depth WebSphere-based application performance analysis and tracing facilities. It provides detailed reports that you can use to enhance the performance of your applications.

ITCAM for WebSphere enables you to analyze the health of the WebSphere Application Server and the transactions that are invoked in it. It can trace the transaction execution to the detailed method-level information. It connects transactions that spawn from one application server to another. It also invokes services from other application servers, including mainframe applications in IBM IMS™ or IBM CICS®. ITCAM for WebSphere provides a flexible level of monitoring, from an non-intrusive production ready monitor, to a detailed deep-dive tracing for problems of locking or even memory leaks. ITCAM for WebSphere provides a separate interactive web console and allows monitoring data to be displayed on the Tivoli Enterprise Portal.

Figure 23 shows the overall architecture of ITCAM for WebSphere.

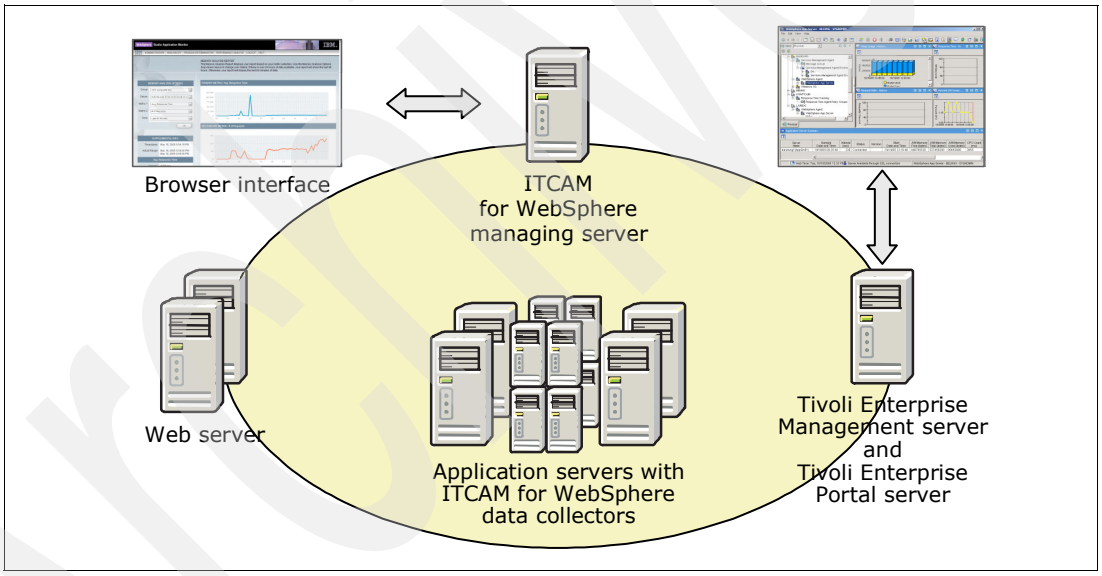

Figure 23 ITCAM for WebSphere architecture

IBM WebSphere Portal Server

IBM WebSphere Portal is a web portal solution that is an integration framework for enterprise information systems. With WebSphere Portal, integration *happens at the glass*. In other words, WebSphere Portal renders several information sources, also known as *portlets*, into one browser window. One or more portlets can be displayed on each page in a hierarchical page structure. Each portlet is like a small browser window, without the control buttons, that displays its specific HTML content. The WebSphere Portal Server integrates each portlet's content into one unified web site structure.

IBM Business Process Management

IBM Business Process Manager is a platform that helps customers to automate real-life business processes, including human interaction tasks, web interface actions, and business rule-based decisions. Processes are controlled by a state machine engine or a simple business process. The building blocks of these process steps are stand-alone services, based on the Service Component Architecture (SCA) standard.

WebSphere Application Server V 7.0 included a Feature Pack for Service Component Architecture (SCA). This Feature Pack is now an integral part of the WebSphere Application Server V8 that allows developers to construct applications based on the Open SCA specification. The SCA modules of Business Process Manger and WebSphere ESB use import and export bindings to interoperate with Open SCA services that are running in WebSphere Application Server.

The team who wrote this paper

This paper was produced by a team of specialists from around the world working at the International Technical Support Organization, Raleigh Center.

Alan Corcoran is an IT Specialist working with WebSphere Education since 1996. He has 15 years of experience developing and delivering education courses on WebSphere products. His focus is WebSphere Application Server administration and WebSphere Voice Response administration. He has a bachelors degree in Computer Science and Urban Planning from the State University of New York at Albany. He works and resides in Newport, Rhode Island.

Balazs Csepregi-Horvath is a Customer Engineer working for IBM Global Services in Hungary. He has a bachelor's degree in Heavy Current Automation and a bachelor's degree in Information Technology, both from Kando Kalman Technical College, Budapest, Hungary. He has ten years of expertise supporting WebSphere, Rational, and Information Management products. He is an authorized WebSphere and certified Portal instructor. He also acts as a Project Manager in IBM Hungary.

Addison Goering is a Certified IT Specialist working since 1998 with WebSphere Education. He has over twelve years of experience developing and delivering administrative courses in the entire WebSphere portfolio, including WebSphere Application Server, WebSphere Enterprise Service Bus, and WebSphere Process Server. His area of expertise is in course design and development.

Jose Pablo Hernandez is a WebSphere Software Specialist working since 2007 in GBM Costa Rica. GBM Costa Rica is an IBM Business Partner and exclusive distributor of IBM products in Central America and the Dominican Republic. He has a bachelor's degree in System Engineering from Universidad Latina de Costa Rica. His areas of expertise include WebSphere Application Server projects implementation and support. He is also an IBM Certified Advanced System Administrator.

Julien Limodin is an IT Specialist working since 2006 at IBM France. He is currently working on customer performance benchmarks, especially on WebSphere and Power in the Products & Solutions Support Center in Montpellier (PSSC), and is part of the EMEA benchmarks center. His main areas of expertise are design, implementation, performance tuning, and high availability of WebSphere Application Server environment. He has a MSC degree in Information Technology and Management from a French engineering school.

Sergio Pinto is an IT Specialist working for Integrated Technology Delivery, Server Systems Operations, in Brazil. He has worked at IBM for 15 years. His area of expertise includes support in WebSphere MQ and WebSphere Message Broker on distributed and z/OS environments and support in WebSphere Application Server for z/OS. He has also worked on developing software using COBOL/CICS command level and Visual Basic using the Oracle database. He has a Bachelor's degree in Business Administration and Accounting from Instituto Catolico de Minas Gerais, Brazil.

Thanks to the following people for their contributions to this project:

Margaret Ticknor, Carla Sadtler, Tamikia Barrow, Debbie Willmschen, Stephen Smith
International Technical Support Organization, Raleigh Center

Michael Cheng
IBM Austin, US

Jim Knutson
IBM Austin, US

Thanks to the authors of the previous editions of this paper:

- *WebSphere Application Server V6.1: Technical Overview*, REDP-4191
- *WebSphere Application Server V7: System Management Technical Overview*, REDP-4569

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Obtain more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Stay connected to IBM Redbooks publications

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks
- Follow us on Twitter:

  http://twitter.com/ibmredbooks
- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806
- Explore new IBM Redbooks® publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm
- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

Notices

This information was developed for products and services offered in the U.S.A.

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

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

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

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

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

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

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

COPYRIGHT LICENSE:

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

This document REDP-4756-00 was created or updated on August 3, 2011.

Send us your comments in one of the following ways:
- Use the online **Contact us** review Redbooks form found at:
  ibm.com/redbooks
- Send your comments in an email to:
  redbooks@us.ibm.com
- Mail your comments to:
  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400 U.S.A.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

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

AIX®	Jazz™	Redbooks (logo) ®
CICS®	Parallel Sysplex®	System i®
ClearCase®	Rational Team Concert™	System z®
DataPower®	Rational®	Tivoli®
IBM®	Redbooks®	WebSphere®
IMS™	Redpaper™	z/OS®

The following terms are trademarks of other companies:

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

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

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