IBM Software Group

IBM WebSphere Application Server v6

Overview

IBM Confidential

Updated October 12, 2004

Agenda

- Main themes
- Product packaging
- Architectural overview
- New features overview

Section

Main Themes & Product Packaging

Main Themes

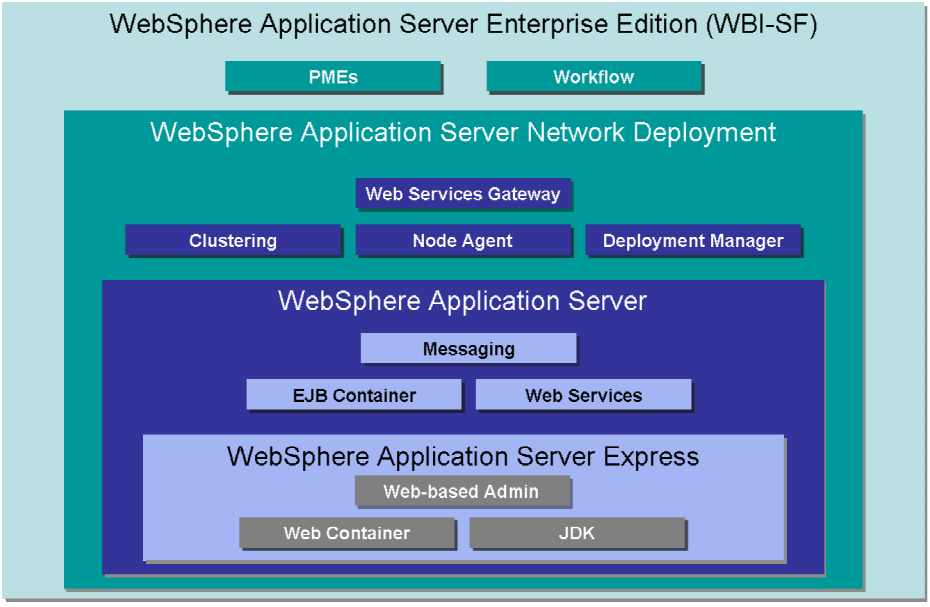
- Platform enablement
  - Developing the e-Business “operating system” for enterprise integration
  - Foundation for other middleware products
- Ease of use
  - Reduced complexity, and increased integration of roles
    - Simplify development and deployment of applications with new WebSphere Rapid Deployment
    - Improvement on the flexible and open Systems Management model from WebSphere v5, with many new enhancements

Main Themes (continued)

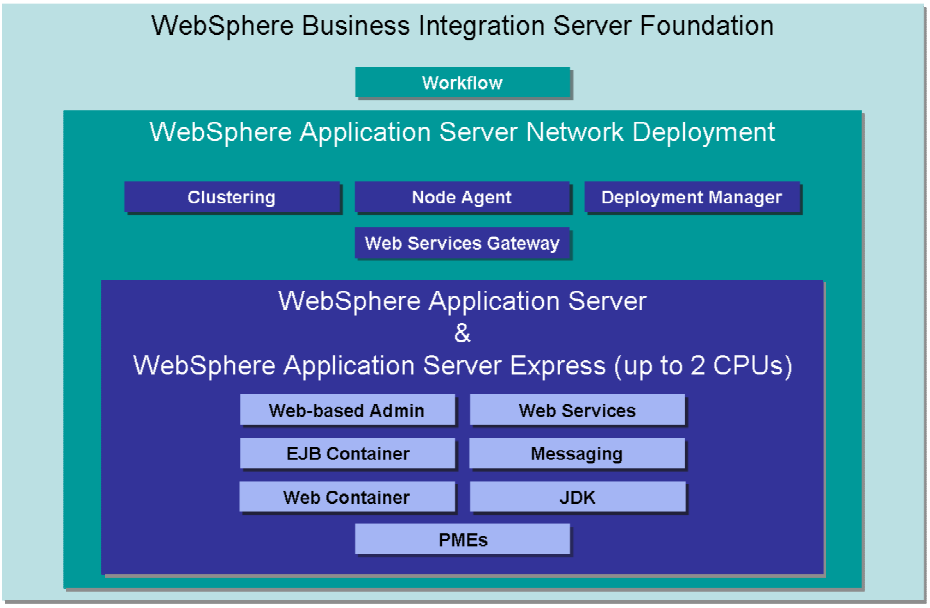
- Enterprise Class Deployment
  - Building the "Next Generation" application server with the qualities of service required by enterprise class deployment
  - Enterprise Service Bus (ESB) infrastructure integration in the Application Server
    - Unifies Service Oriented Architecture synchronous and asynchronous messaging, message brokering and publish/subscribe, mediation and Web Services integration
  - Unified Clustering Framework and High Availability services

- Standards-based architecture and programming model
  - Builds on the latest Java standards and Web Services in an integrated development and deployment environment to reduce time to value
  - J2EE 1.4, Web Services, Service Data Objects (SDO), etc.

Version 5 Packaging



Version 6 Packaging

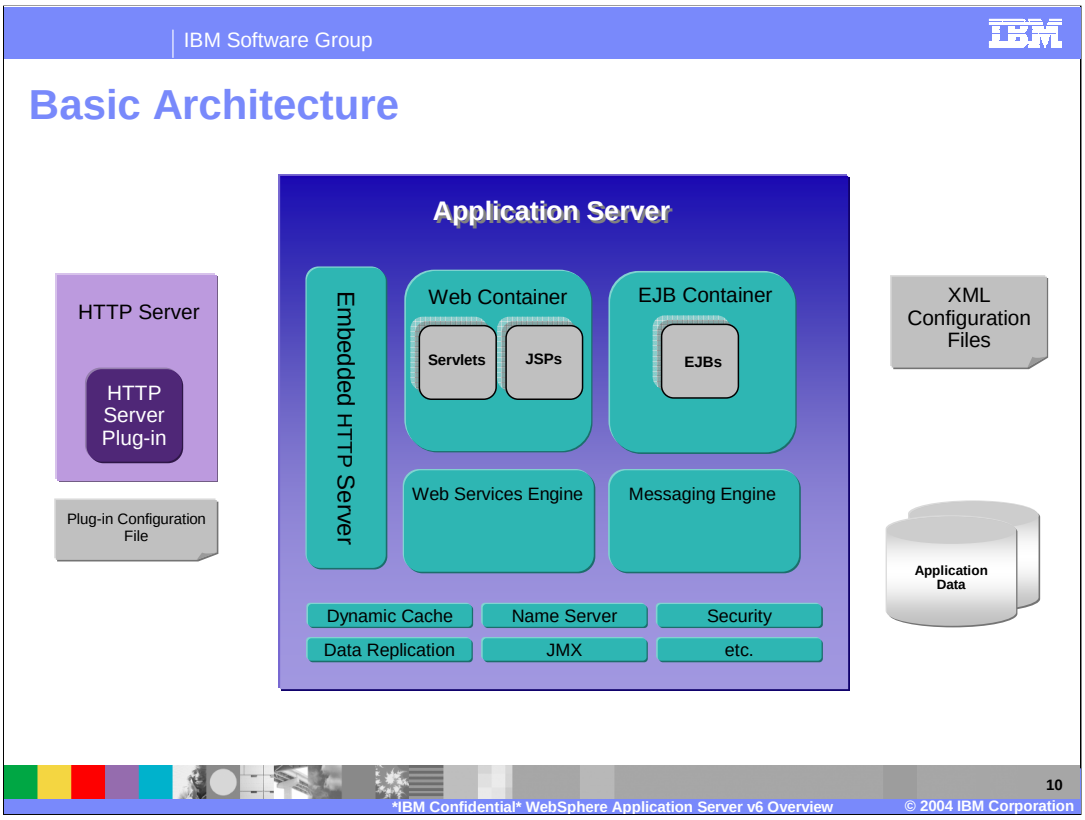


Section

Architectural Overview

WebSphere Application Server Basics

- WebSphere Application Server is a platform on which you can run Java-based business applications
- It is an implementation of the Java 2 Enterprise Edition (J2EE) specification
- It provides services (database connectivity, threading, workload management, etc.) that can be used by the business applications



This diagram illustrates the basic architecture of WebSphere Application Server, including several of the larger components.

The main element is the application server, a Java process that encapsulates many services, including the containers, where business logic executes. If you are familiar with J2EE, you will recognize the Web Container and the EJB container. The Web Container executes Servlets and JavaServer Pages (JSPs), both of which are Java classes that generate markup to be viewed by a web browser. Traffic into and out of the Web Container travels through the embedded HTTP Server. While Servlets and JSPs can act independently, they most commonly make calls to Enterprise Java Beans (EJBs) to execute business logic or access data. EJBs, which run in the EJB container, are easily reusable Java classes. They most commonly communicate with a relational database or other external source of application data, either returning that data to the Web Container or making changes to the data on behalf of the Servlet/JSP.

As you may have noticed, the v6 application server now has a JMS messaging engine built into the application server. This is a pure-Java messaging engine, unlike the embedded version of WebSphere from MQ that was part of version 5. JMS destinations, known as queues and topics provide asynchronous messaging services to the code running inside the containers. JMS will be covered in more depth later in this course.

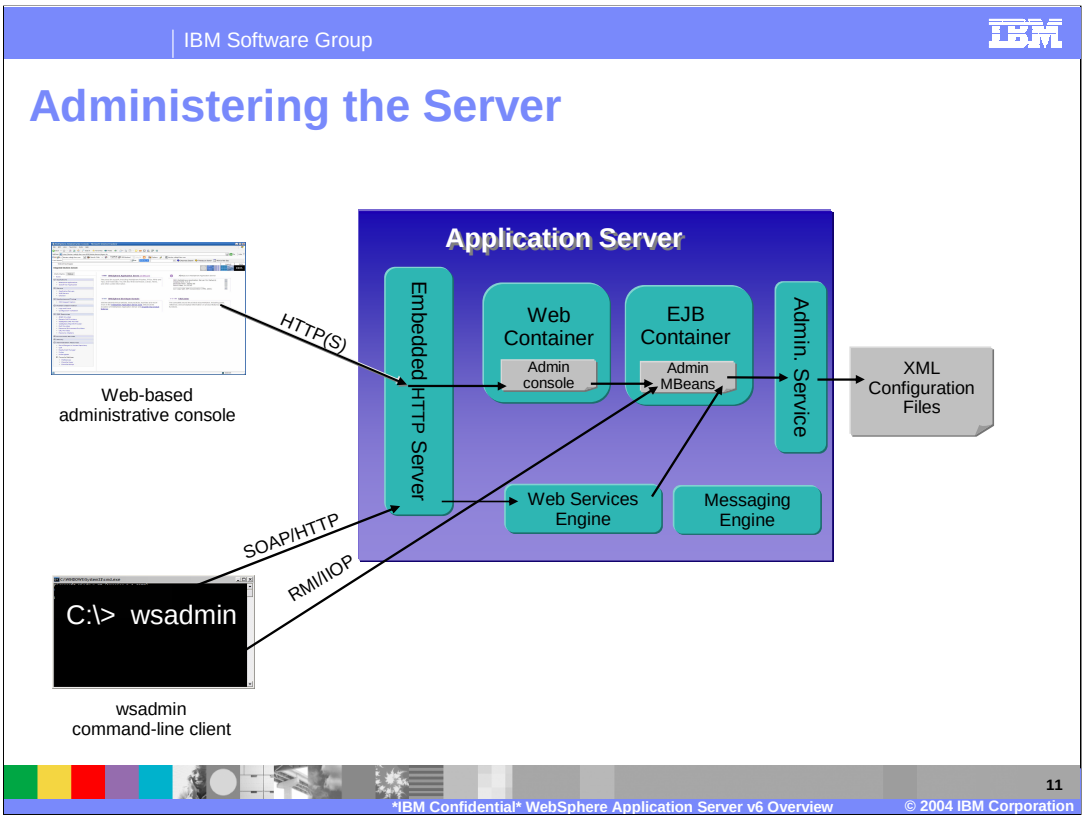
As you will see in more detail later on, the Web Services engine enables application components to be exposed as web services, which can be accessed using Simple Object Access Protocol (SOAP).

Several other services run within the application server, including the Dynamic Cache, Security, and others. These will be covered later in the class, as they are typically more advanced.

There are also some important components outside of the application server process.

The server's configuration is stored in a set of XML files, often referred to as the configuration repository. These files define the server itself, as well as resources and services that it provides.

WebSphere Application Server also provides a plug-in for HTTP servers that determines what HTTP traffic is intended to be handled by WebSphere, and routes the requests to the appropriate server. The plug-in is also a critical player in workload management of HTTP requests, as it can distribute the load to multiple application servers, as well as steer traffic away from unavailable servers. It too reads its configuration from a special XML file.



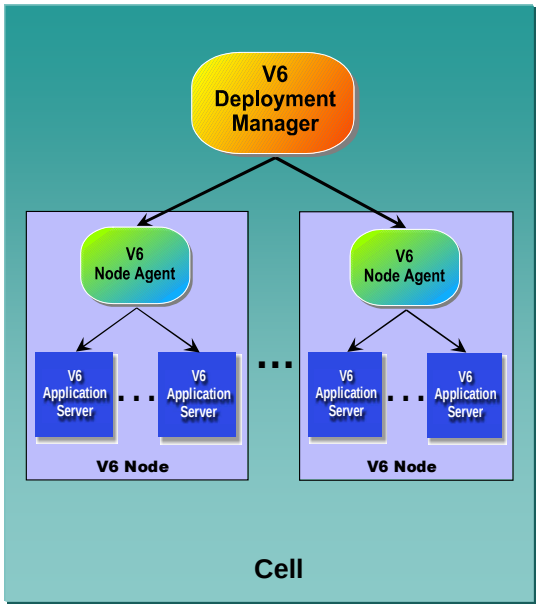
The most common way to control an application server is via the web-based administrative console. This web interface enables you to configure and control all aspects of an application server, from installing and managing applications to configuring diagnostic tracing. The administrative console is a web application that runs inside the web container. This application makes calls to Management EJBs (MBeans), which communicate with the admin. service. The admin service then manipulates the XML configuration files to reflect the changes made by the web client.

Another common way to administer the server is using the command-line scripting client, wsadmin. Wsadmin is a command interpreter that can be used interactively or in batch mode. It can communicate with the MBeans directly (using RMI/IIOP) or via the web services engine (using SOAP/HTTP), to perform most of the functions that can be performed using the Administrative Console.

It is also possible to write your own custom administrative client with the JMX API if you have particular needs that are not filled by wsadmin or the web interface.

Network Deployment Concepts

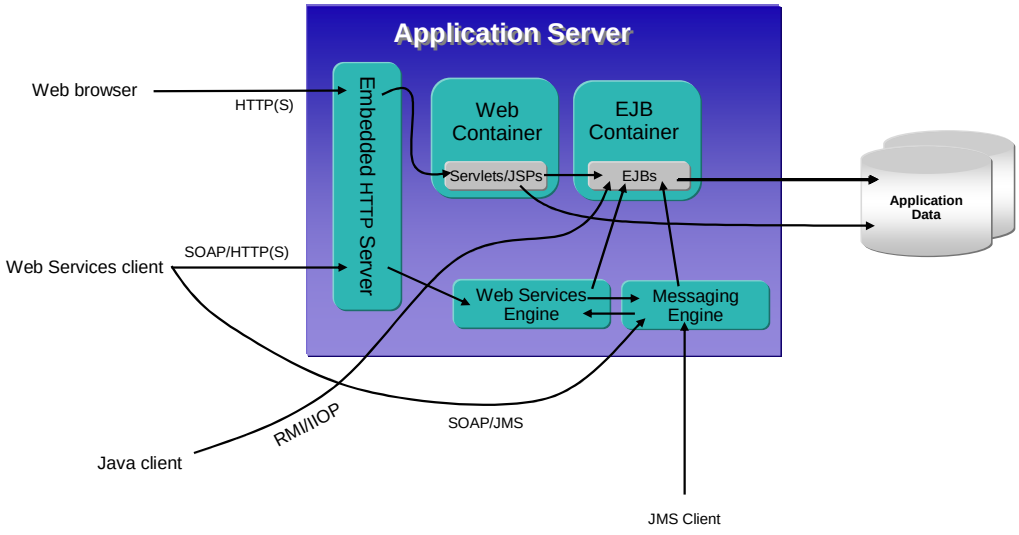
- A node is a logical grouping of servers
  - Each node is managed by a single node agent process
- A deployment manager process manages the node agents
  - Holds the configuration repository for the entire management domain, called a cell
  - Administrative Console runs inside the DMgr



Managed vs. Unmanaged Nodes

- A managed node is a node that contains a node agent
  - Used to be called a “node” in v5
- An unmanaged node is defined in the topology, but does not have a node agent process
  - Enables the rest of the environment to be aware of the node
    - Useful for defining HTTP servers as part of the topology, enabling creation of different plug-in configurations for different HTTP servers
  - Lack of a node agent prohibits any management by WebSphere

Accessing Server Resources



Section

WebSphere Profiles

WebSphere Profiles Overview

- WebSphere files are split into 2 components:
  - Product Files
    - Set of shared read-only static files or product binaries shared by any functional instance of the WebSphere Application Server product
  - User Files
    - Set of user-customizable data files. This set of files is called a **profile**
    - User data includes WebSphere configuration, installed applications, resource adapters, properties, log files, etc.
- Benefits of profiles:
  - Each Profile provides a WebSphere runtime environment sharing the same product binaries
    - WebSphere runtime environments: Stand-alone Node, Managed Node and DMgr
  - Easier than multiple installations
    - Less disk space
    - Simplifies application of product updates

Section

New Features Overview

New Features Overview

- Programming models
- System Management
- WebSphere Rapid Deployment
- Platform Messaging
- Workload Management – High Availability
- Security

Section

Programming Models

Application Programming Model Support

- Supported J2EE Application versions in v6 Application Server
  - New v6 J2EE 1.4
  - J2EE 1.3
  - J2EE 1.2
- Simplifies migration by allowing your existing J2EE 1.2 or 1.3 applications to run in v6 Application Server

Similar to v5 Application Server, you will have v4 Data Source for your J2EE 1.2 applications needing access to back end Relational database

J2EE 1.4 Overview

Web Services and XML support

- **Standards / Portability** - XML Schema definitions for all deployment descriptors
- **JAX-P 1.2** - New properties for XML parsers
- **JAX-R** - XML registry API
- **JAX-RPC** - APIs for representing WSDL-based services as RPCs in Java (and vice-versa)
- **JSR 109** - Web services programming and deployment model
- **SAAJ 1.1** - SOAP Attachments API for Java

Pluggable Messaging

- **EJB 2.1**
  - Typed message beans (used for any inbound JCA including pluggable JMS provider)
  - Timer service Web service end-point support
- **JMS 1.1**
  - Unification of point-to-point and pub-sub interfaces
- **J2CA 1.5**
  - In-bound connections (supporting pluggable JMS provider, generalized for other types)
  - RA lifecycle support
  - Work manager (threads for resource adapters)

ISV Enablement

- **JMX 1.2 / JSR-077 (J2EE Management)**
  - Notification emitters, and standard patterns
  - Information model representing J2EE application server concepts
- **JSR-088 (J2EE Deployment)**
  - XML-based deployment interfaces for J2EE
- **JACC 1.0**
  - Java Authorization Contract with Containers
  - APIs for registering J2EE component authorization policies

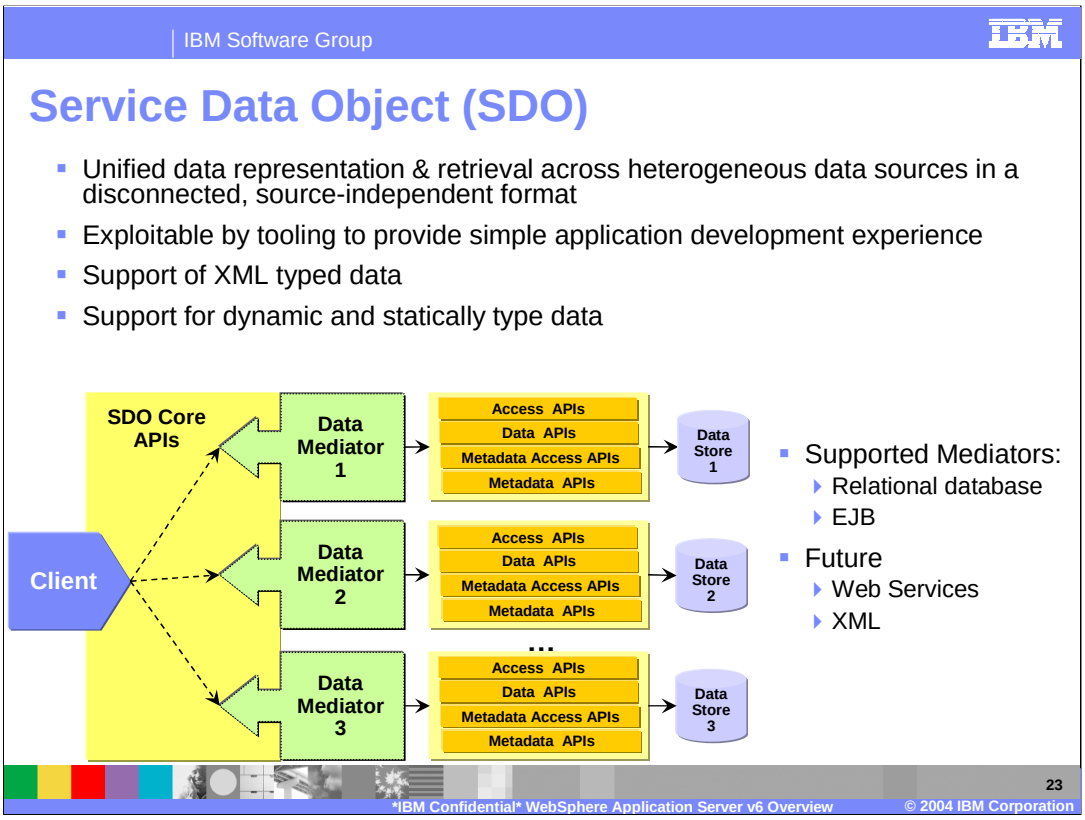
Other

- **Servlet 2.4**
  - Extensible deployment descriptors
  - Request/response listeners
- **JSP 2.0**
  - Expression Language
  - Simple Tag Extension
- **JDBC 3.0**
  - Meta data and cursor support
- **JavaMail 1.3** updates

http://java.sun.com/j2ee/j2ee-1_4-fr-spec.pdf

Changes in Web Services

WebSphere 4.0 & 5.0	WebSphere 5.02/5.1	WebSphere 6.0
Apache SOAP ▪ The programming model, deployment model and engine **Proprietary APIs** ▪ Because Java standards for Web services didn't exist **Not WS-I compliant**	**JAX-RPC (JSR-101) 1.0** ▪ New standard API for programming Web services in Java **JSR-109 1.0** ▪ New J2EE deployment model for Java Web services **SAAJ 1.1** **WS-Security** ▪ Extensions added **WS-I Basic Profile 1.0** ▪ Profile compliance **UDDI4J version 2.0 (client)** **Apache Soap 2.3 enhancements** The engine is a new high performance SOAP engine supporting both HTTP and JMS	**JAX-RPC (JSR-101) 1.1** ▪ Additional type support ▪ xsd:list ▪ Fault support ▪ Name collision rules ▪ New APIs for creating Services ▪ isUserInRole() **JSR-109 - WSEE** ▪ Moved to J2EE 1.4 schema types ▪ Migration of web services client DD moving to appropriate container DDs ▪ Handlers support for EJBs ▪ Service endpoint interface (SEI) is a peer to LI/RI **SAAJ 1.2** ▪ APIs for manipulating SOAP XML messages ▪ SAAJ infrastructure now extends DOM (easy to cast to DOM and use) **WS-Security** ▪ WSS 1.0 ▪ Username Token Profile 1.0 ▪ X.509 Token Profile 1.0 **WS-I Basic Profile 1.1** ▪ Attachments support **JAXR support** **UDDI v3 support** ▪ Includes both the registry implementation and the client API library ▪ Client UDDI v3 API different than JAXR (exposes more native UDDI v3 functionality)



The primary goal of the SDO architecture is to make it easier for application and tools developers to create, view, update, and delete data that is stored in a variety of backend data stores. One of the reasons this is currently a challenge is that there are a wide variety of APIs and data models that are commonly used for J2EE application development. The SDO architecture addresses this problem by providing uniform data access and representation across a wide variety of data sources as well as support for many common application patterns that are encountered in J2EE application development. The intension is to decrease the amount of low level code developers need to write in order to create an application, and instead focus on solving the business problem.

Goals and Benefits:

•Rapid, simple application development

- •Current J2EE programming models are too complex
- •Exploited by Tooling

•Data-source independent data access

- •Today, there are many different models and APIs for data access in J2EE
- •A single model reduces complexity

•Data-centric data access

- •No behavior associated with data

•XML integrated

- •Easy to transfer data between tiers/Web Services

•Disconnected Model

- •Normal mode of operation for Servlets and JSPs
- •Provides a performance advantage by reducing database round-trips

SDO vs. JDO

SDO is more generic than JDO. SDO deals with data flow in addition to persistence issues. You could even conceivably use SDO mediators to access JDO data. For more information, see http://www-106.ibm.com/developerworks/java/library/j-sdo/

JavaServer Faces (JSF)

- Provide an easier and visual way to build J2EE Web applications with rich set of UI for a variety of client devices
- WebSphere Application Server v6 runtime and IBM Rational Web/Application Developer tools
  - Supports JSF v1.0
  - JSF jar files and tag libraries are included with the runtime environment
  - Includes a number of IBM value-add JSF custom components, permitted by the specification

In WebSphere Studio Application Developer v5.1.1 and WebSphere Application Server 5.1, JSF was supported as a Beta technology. However, during this Beta offering, JSF jar files and tag libraries were not included with the WebSphere Runtime environment. Prior to v6, JSF jar files and tag libraries need to be packaged with a JSF application. However, starting with version 6, JSF jar files and tag libraries will be included with the runtime environment.

Programming Model Extensions

- Programming model extensions (PMEs) are IBM-developed extensions to the J2EE model
- Core extensions included in all versions
  - Formerly available only in Enterprise Edition

•Last Participant Support
•Internationalization Service
•WorkArea Service
•ActivitySession Service
•Extended JTA Support
•Startup Beans
•Asynchronous Beans (now called WorkManager)
•Scheduler Service
•Object Pools
•Dynamic Query
•Web Services Gateway Filter Programming Model (with migration support)
•DistributedMap
•Application Profiling

Section

System Management

System Management Features

- Extends v5 System Management Model
  - Reduces learning curve for managing v6 environments
- Support for J2EE 1.4 specification
  - JMX 1.2
  - J2EE Management (JSR-077)
  - J2EE Deployment (JSR-088) features

New v6

- Fine-grained Application Update
  - Ability to introduce small delta-changes to installed applications
  - Ability to add, update or remove parts of the installed application and restart the changed part

New v6

- Support for extensible Server types
  - Web Server
  - Generic Server

New v6

- Introduction of Node Groups
  - z/OS and Distributed Nodes within same cell

New v6

•System Management is build on v5 model with the configuration files in XML

•WebSphere Instance enhancements

- •New wsInstance architecture, now called Server profiles
- •Many new options to manage instances
- •Default Application Server and Deployment Manager are also an instance

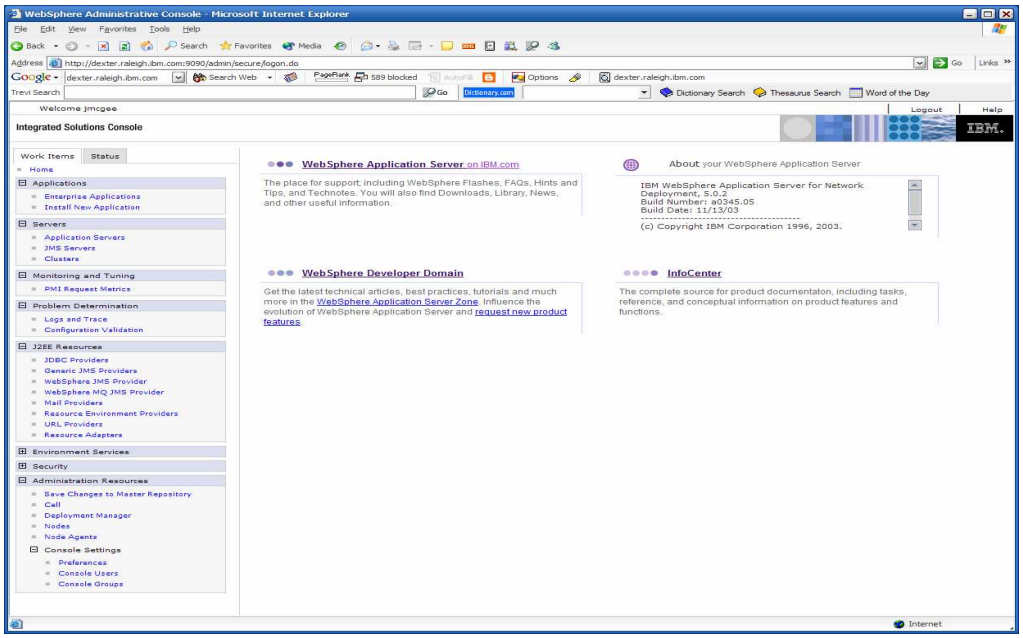
System Management Features (continued)

- Improved Administrative Console look and feel
  - Provides consistent cross-IBM product look and feel
  - Changes console views based on context
    - Version
    - Platform
    - Installed Capabilities
  - Integration of Tivoli Performance Viewer
  - New v6 Integration of IBM HTTP Server v6 management
- New v6 "System applications": Administrative Console, File Synchronization

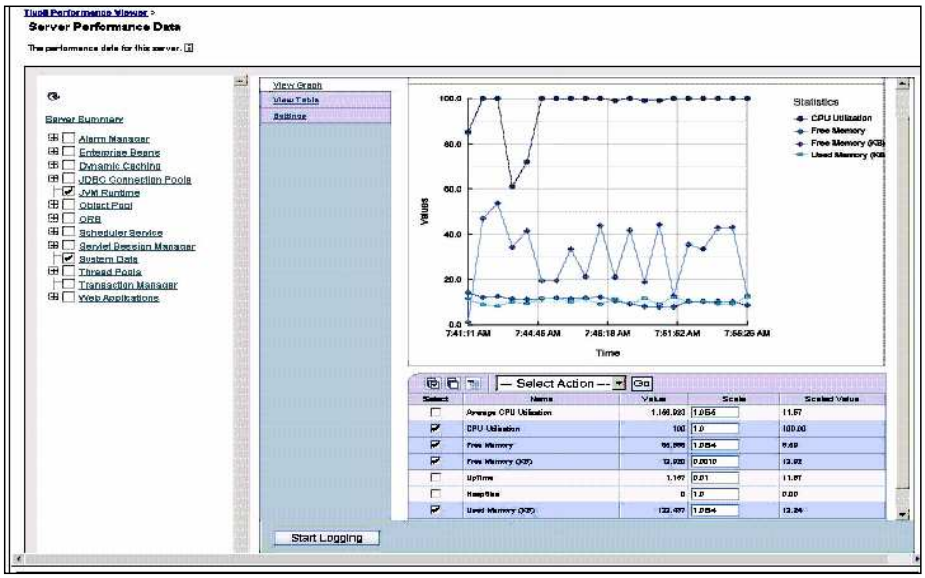
WebSphere Configuration Archive

- Basically the same as a regular WebSphere configuration, with two main differences:
  - It may be a subset of a full configuration
  - Configuration information is virtualized to make it portable
    - Removes any specific information, like the host name
- WebSphere Configuration Archives are used to import/export configurations
- Allows simple creation of many servers with the same configuration

New Admin Console Look and Feel



Integrated Performance Viewer



Section

WebSphere Rapid Deployment

WRD: Deployment Automation

- Speeds and simplifies development and testing
- Two styles supported:
  - Automatic installation of applications and modules onto a running WebSphere Server - local or remote servers
  - Free form application development - initially supported in Headless mode only
    - Enables a “Hot Directory” concept for “file copy” and “Notepad” development and deployment using annotations
    - Supports Automated EAR management or by-part application construction
    - Makes key decisions about default settings
    - By-part application construction allows you to dump your artifacts in a directory – does not have to follow J2EE packaging

Monitors a directory within workspace through Rational Application Developer (RAD) or with Eclipse User-Interface (UI)

WRD: Annotation-based Programming

- Developer adds metadata tags into application source code
  - Uses XDoclet tag syntax, where defined
- WRD uses the metadata to generate additional artifacts needed to run the application on the Application Server
- Minimizes number of artifacts a developer needs to create and understand – user maintains the single artifact

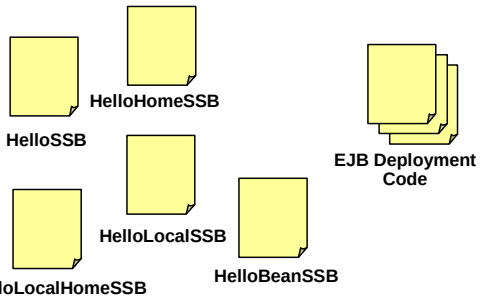
Single Java Source File with Annotation-based programming

```
package com.ibm.wrd;
  /**
   *   @ejb.bean name="Hello" type="Stateless"
   * view-type=both  jndi-name="HelloBean"
   */
 public class Hello
 {
    /**
    * @ejb.interface-method view-type=both
    */
     public String hello(String name)
     {
        return "Hello: " + name;
 }
}
```

Hello.java

Generates

Multiple Java Source Files and application artifacts



Section

Platform Messaging

Platform Messaging

- WebSphere Application Server Platform Messaging is a fully-integrated pure-Java messaging environment
- Its allows WebSphere Application Server to participate in an implementation of an Enterprise Service Bus
  - ESB is a conceptual model for SOA
  - SIBus is the WebSphere implementation of a single ESB component
- JMS support is built on top of platform messaging

The service integration functionality within WebSphere Application Server provides a highly-flexible messaging fabric that supports a service-oriented architecture with a wide spectrum of quality of service options, supported protocols, and messaging patterns. It supports both message-oriented and service-oriented applications.

WebSphere Platform Messaging in v6

- Platform messaging fully integrated within WebSphere Application Server
  - Integrated with WebSphere Security
  - Common install process
  - Fully Integrated with WebSphere System Management
    - Admin Console provides MQ-Explorer type management
  - All Java implementation within the server process - No external processes
    - Co-exist with WebSphere MQ
  - Performance monitoring, Trace and Problem Determination

Improved performance for in-process messaging

Messages traveling between applications running in the application server process can be passed in memory

Section

Workload Management – High Availability

Unified Clustering

- Management consistency for clustering of different resources
  - Operational ease of use - The view and use of clusters will be administered in a unified and consistent manner for all protocols (HTTP, EJB, JMS, JCA, etc)
- Consistency - New WLM functions (weighted distribution, eWLM integration, SLA, hardware provisioning, etc.) are implemented once for all protocols
- High Availability - Makes WLM a highly available service which make cluster and routing information always available

Data Replication Service Enhancements

- Integrated with High Availability Manager
  - Improved performance and scalability
    - Provides a more optimized communication stack
    - Allows for use of both unicast and multicast IP
    - Improves in the range of 4x to 8x
  - Improves high availability and failure recovery:
    - Leverages the failure detection provided by high availability services
    - Along with the WLM / Unified Clustering integration, this allows for "active failure recovery"
      - For example, with HttpSession replication, if the affinity server for a HttpSession goes down, WLM can route to another server that has a backup copy ready to use
  - Improves usability:
    - Leverages group services to simplify partitioning
      - Now have "n-replica", where the customer simply defines the number of backup copies they want for data
- Stateful Session Beans state now replicated

What's new?

- The v6.0 of the Application Server incorporates an improved version of the Data Replication Service that has now been rebased on top of a multicast based transport, the High Availability services and the Channel Framework.

- As a result of this change, the Data Replication Service will be simpler to configure and use, will perform and scale better. The product will also be simpler to configure with scripting, since a lot of the complicated tuning options are now unnecessary.

- Integration with WLM will allow WLM to intelligently pick a backup application server to route subsequent http requests to, such that future requests will be sent to an application server where a backup session exists.

- Added users of this component include Stateful Session Beans failover and the Entity Bean Persistence manager.

- Enhanced features and improved performance will be available to all the users of DistributedMap as well.

What used to happen?

- v5.0 of the Application Server shipped with a JMS based transport, that was not as highly performing.

- In addition it was not completely intuitive how to configure this component and there was a fair amount of complexity involved it setting it up.

Why does the customer care?

- Firstly it will be much simpler and more intuitive for the customer to configure Data Replication Service. More sample scripts will make it easier to use wsadmin to setup the component.

- The performance of the product will also be improved.

Top 3 points of this slide?
1. Improved performance and scalability
2. Active failure recovery
3. Stateful Session Beans and Entity Bean Persistence Manager cache now replicated

Section

Security

Security Enhancements

- Java Authorization Contract with Containers (JACC) 1.0 support
  - Allows plug-in of your Authorization servers
  - JACC compliant TAM (Tivoli Authorization Module) shipped with v6
  - Will continue to support the non-JACC native authorization (similar to v5)
- Security Attribute Propagation from WebSphere Application Server v5.1.1
- Implements WS-Security 1.0

Section

Summary and Reference

Summary

- Introduced high level architecture of WebSphere Application Server v6
  - Requests can come into the application server in several ways
    - The same business logic can be wrapped in a Servlet, WebService, or invoked directly
- Version 6 supports the J2EE 1.4 specification
- WebSphere profiles allow several server instances to share the same set of product binaries

Summary (continued)

- WebSphere Application Server v6 provides many new functional enhancements
  - Programming model (J2EE, Web Services, SDO, JSF, and Programming Model Extensions)
  - System Management
  - Simplified development and deployment
  - WLM and High Availability
  - Security

Trademarks and Disclaimers

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

IBM	iSeries	OS/400	Informix	WebSphere
IBM(logo)	pSeries	AIX	Cloudscape	MQSeries
e(logo)business	xSeries	CICS	DB2 Universal Database	DB2
Tivoli	zSeries	OS/390	IMS	Lotus

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both. Microsoft, Windows, Windows NT, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both. Intel, ActionMedia, LANDesk, MMX, Pentium and ProShare are
trademarks of Intel Corporation in the United States, other countries, or both. UNIX is a registered trademark of The Open Group in the United States and other countries. Linux is a registered trademark of Linus Torvalds. Other company, product and service names may be trademarks or service marks of others.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or program(s) described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (e.g., IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. 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 in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

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

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.