75

MQ update

September 2005

In this issue

A tcl PUBLICATION

MQ Update

Published by

Xephon Inc
PO Box 550547
Dallas, Texas 75355
USA

Phone: 214-340-5690
Fax: 214-341-7081

Editor

Trevor Eddolls
E-mail: trevore@xephon.com

Publisher

Colin Smith
E-mail: info@xephon.com

Disclaimer

Readers are cautioned that, although the information in this journal is presented in good faith, neither Xephon nor the organizations or individuals that supplied information in this journal give any warranty or make any representations as to the accuracy of the material it contains. Neither Xephon nor the contributing organizations or individuals accept any liability of any kind howsoever arising out of the use of such material. Readers should satisfy themselves as to the correctness and relevance to their circumstances of all advice, information, code, JCL, scripts, and other contents of this journal before making any use of it.

Subscriptions and back-issues

A year's subscription to *MQ Update*, comprising twelve monthly issues, costs $380.00 in the USA and Canada; £255.00 in the UK; £261.00 in Europe; £267.00 in Australasia and Japan; and £265.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the July 2000 issue, are available separately to subscribers for $33.75 (£22.50) each including postage.

Contributions

When Xephon is given copyright, articles published in *MQ Update* are paid for at the rate of $160 (£100 outside North America) per 1000 words and $80 (£50) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of $32 (£20) per 100 lines. To find out more about contributing an article, without any obligation, please download a copy of our *Notes for Contributors* from www.xephon.com/nfc.

MQ Update on-line

Code from *MQ Update,* and complete issues in Acrobat PDF format, can be downloaded from our Web site at www.xephon.com/mq; you will need to supply a word from the printed issue.

Printed in England.

WebSphere MQ V6: what's it all mean?

WebSphere MQ is IBM's award-winning middleware for commercial messaging and queueing. It is used by thousands of customers in every major industry in many countries around the world. It ensures reliable delivery of data and frees application developers from the task of building complex recovery and error handling logic into their applications. The WebSphere MQ products enable programs to communicate with each other across a network of unlike components, such as processors, subsystems, operating systems, and communication protocols.

WebSphere software includes business integration products and solutions that can help integrate just two applications, a single department, or the entire enterprise, reaching across business channels, trading partners, and Web customers. The WebSphere platform includes:

- WebSphere MQ, the core of application integration, which integrates many platforms. Providing the messaging foundation for an enterprise service bus and ensuring reliable message delivery, it can be used alone or combined seamlessly with WebSphere Application Server. Both WebSphere MQ and WebSphere Application Server provide messaging resources that can form the foundation of a company's ESB, which can grow incrementally with their business needs.
- WebSphere MQ Extended Security Edition, which enhances WebSphere MQ with end-to-end application-level data protection features. It enables enterprise-wide, remote management of security policies on your MQ network and can be deployed to existing production environments without changes to existing WebSphere MQ applications.
- WebSphere Business Integration Message Broker, a powerful information broker that includes a one-to-many

connectivity model plus transformation, intelligent routing, and information flow modelling across multiple disparate business systems. It also supports publications and subscriptions including mobile clients and remote telemetry devices.

- WebSphere MQ Everyplace, which brings the benefits of assured message delivery and rock-solid security to the failure-prone environment of mobile working.
- WebSphere MQ Workflow, a business process management system enabling the definition, execution, and swift change of complete business processes that span systems, applications, and people.
- WebSphere Business Integration Server Foundation builds on the best-in-class WebSphere Application Server Network Deployment to provide a powerful standards-based integration platform for building and deploying Business Process Execution Language for Web Services (BPEL) and Web service-based composite applications.

WebSphere MQ V6.0 areas of improvement include:

- 64-bit interfaces on AIX, HP-UX, and Solaris.
- Internet Protocol Version 6 (IPv6) support.
- Integrated support for Web services data over a reliable MQ transport.
- Integrated publish/subscribe capability.
- An extensible Eclipse-based configuration user interface on Microsoft Windows and Linux *x*86 platforms.
- The product Quick Tour introduces the basic concepts of application connectivity and describes WebSphere MQ functions and capabilities; the File Transfer utility (on Microsoft Windows and the Linux *x*86 platform) provides a simple way of sending data (messages) from one MQ system to another.

- IBM Message Service Client for C/C++ support.
- Manageability: easier problem determination and avoidance, and the ability to monitor the MQ network, gather statistics, and raise alerts.
- Integrated Java Message Service (JMS) V1.1 support.
- IBM Power platform support (Linux for iSeries and Linux for pSeries).
- Connects z/OS applications across more than 80 supported platform configurations: now with support for IBM Power platforms for both Linux for iSeries and Linux for pSeries.
- Provides integrated support for Web services, enabling SOAP requests to flow over the reliable WebSphere MQ backbone helping integrate J2EE, CICS, and .NET applications.
- Interoperates seamlessly with the messaging resources of WebSphere Application Server V6.0 to form the proven basis of your Enterprise Service Bus (ESB).
- Full support for the *de facto* standard API (MQI) and the Java-based standard API (Java Message Service (JMS) V1.1).
- Can be configured remotely using new Eclipse technology-based tooling included with WebSphere MQ V6.0.
- Integrated support for simple configuration of messaging using publish-and-subscribe messaging distribution.
- Supports industry-standard Secure Sockets Layer (SSL) security and offers an Extended Security Edition for advanced security features.
- Enables you to move your existing FTP infrastructure forward, ensuring reliable secure file transfer over WebSphere MQ.
- Workload balancing and communication failover with clustering support.

- Performance, capacity, and scalability.
- Additional new functions.

WMQ V6 NEW FEATURES AND ENHANCEMENTS

You can basically identify eight areas of improvement with this version of WebSphere MQ:

- Manageability
- Usability
- Performance
- Capacity
- Availability
- Scalability
- Standards
- New functions (including ESB).

Let's learn about the changes and how they provide you with a better environment.

The following features and enhancements are included in Version 6.

Manageability

Easier problem determination and avoidance

New status information shows whether messages are being processed or are being delayed and indicates what might be causing any delays.

Statistics similar to the System Management Facility (SMF) data for z/OS systems are written to event queues, enabling better monitoring of application and queue manager behaviour. This could be used for monitoring the required capacity of a system, giving an indication of workloads.

It is easier to see important information returned from status commands because Programmable Command Format (PCF) and MQSC commands now have a WHERE clause that filters queries. This can be used in commands that enquire about status, to see potential problems before they cause a significant outage.

Developers who write WebSphere MQ management and monitoring applications use the PCF interface. This release extends PCF to the z/OS Command Server, which will permit developers to write a single common application to manage both distributed and z/OS systems. PCF has also been enhanced with new data types and the ability to group or nest sets of parameters.

PCF is used as the format for event messages; this release extends the number of different events generated by WebSphere MQ. For example, events are now available from the distributed platforms that describe in-use transaction logs.

Another set of events on the distributed queue managers corresponds to the SMF data generated by the z/OS queue manager. SMF is a z/OS facility that collects and records system and job-related information, which you can then use to create reports for use in managing your installation. Statistical and accounting information showing the work being done by the queue manager and individual applications is written as event messages. The messages can be read by any application and used for purposes such as future capacity planning, charge-back, or looking for consolidation opportunities.

Activity reports

New report options can be put in the message descriptor. These request (but do not require) applications that process the message to generate reports on what they have done with that message. Generically, these are known as activity reports.

The queue manager and channel programs have been enhanced to generate these reports, specifically to show the passage of messages through a queue manager network. Analogous to the TCP/IP trace route command, the WebSphere MQ **dspmqrte** command reports on which transmission queues are used, and which channels are transporting the message. This can be used to test, debug, and monitor a network configuration.

Developers should consider whether the generation of activity reports might be useful for their particular applications. Activity reports are written in PCF format, which now permits arbitrary user-defined parameters to be included in PCF messages.

Monitor the MQ network and change the way it is working

The new **DISPLAY CONN** command shows which applications are connected to a queue manager and which queues they have opened.

The new **STOP CONN** command forces a disconnection from a queue manager, allowing an MQ administrator to disable a problem application.

New attributes for queues, queue manager channels, and cluster channels extend the range of cluster workload algorithms.

WebSphere MQ can track the route that a message would be expected to take through its network. New flags can request that queue managers that process the message should send back an activity report (for example, which channel, queue manager, and transmission queue have been used).

The size of the active log is now 128GB on Unix platforms and 64GB on Windows systems.

Event messages are generated when log files are switched, and it is now easier to automate the copying and archiving of log files. In addition, a queue manager can be made to replay copied logs without fully starting.

Usability

Configuration tooling

WebSphere MQ V6 has a new graphical configuration tool replacing the Graphical User Interface (GUI) that was part of WebSphere MQ V5.3, and which was restricted to running in the Microsoft Management Console. You no longer need a Windows machine for the GUI; the WebSphere MQ V6 implementation is based around the Eclipse framework and is supported on both Windows and Linux. Moving to Eclipse provides many advantages, including the ability to run on multiple operating systems, and the potential integration with other products that are also using Eclipse for application development or management.

As shipped, the WebSphere MQ V6 GUI repeats the function from the Microsoft Management Console (MMC version) and then extends it in a number of ways. One of these extensions is that it is now possible to manage a z/OS queue manager. The queue manager being managed does not have to be running WebSphere MQ V6 except on z/OS; older versions of WebSphere MQ and MQSeries on distributed platforms are also manageable from this tool.

Eclipse is an award-winning open source platform for the construction of powerful software development tools and rich desktop applications.

Extendability of the Eclipse configuration tooling

The GUI has been implemented as Eclipse plug-ins and can itself be extended by writing new plug-ins. The Eclipse interface for plug-ins includes both methods that are common to all plug-ins and a standard way in which each plug-in can expose additional functions. The WebSphere MQ GUI includes documented interfaces that permit third parties to add their own items to menus, such as those that appear when you press the right mouse button when looking at a queue. Sample plug-ins are provided in both source and binary formats showing how the standard Eclipse methods are followed.

Quick Tour

A product, Quick Tour, introduces the basic concepts of application integration and describes WebSphere MQ functions and capabilities.

Reliable file transfer

WebSphere MQ reduces the time and skills required to be productive with a new file transfer application that provides a simple GUI tool to move files across the MQ network reliably. In addition, simple scripting commands for file transfers enable batches of files to be transferred, triggered by the operating system or by applications.

Performance, capacity, availability, and scalability

64-bit support

In this release, the queue manager has become a 64-bit program on Unix platforms (AIX, Solaris, and HP-UX). This provides improved scalability within the queue manager because much more data about queues and messages on those queues can be held in memory.

Changing the queue manager does not directly affect application developers. Developers can now choose whether to build applications as 32- or 64-bit. MQI libraries for both client and local binding connections are shipped in both formats. In most cases, we expect applications to remain at 32-bit. There is no need to recompile existing applications when you move to WebSphere MQ V6 because there is binary compatibility moving forward, but some applications might want to take advantage of the extended address space that is now available.

There is no significant performance difference when comparing the connection of 64-bit applications and 32-bit applications with a WebSphere MQ V6 queue manager; any performance improvements would come from redesigning applications. However, there is an impact on anyone who writes exits that are invoked by applications or the queue manager. All the various exits, including channel exits, data conversion, and API exits, that run on these Unix platforms need to be compiled and linked in both 32- and 64-bit modes. The product documentation shows how these are built, installed, and configured so that WebSphere MQ automatically picks the correct ‘size’ at run time.

Data compression on channels

All channel types now support compression, which can improve network bandwidth usage and may also improve performance of encrypted SSL channels.

Enhanced availability

WebSphere MQ for z/OS Version 6.0 delivers enhancements to further increase availability by reducing constraints and enabling more configuration changes to be made on-the-fly. For shared queues, the maximum message size has been increased from 63KB to 100MB, the same as for other queues. You can modify more system resources online than before, including page sets, buffer pools, and buffers, as well as channel initiator (CHIN) configuration parameters and SSL configurations. You can also dynamically expand page sets up to 64GB before they become full, and you can dynamically add and remove them, along with buffer pools. The log extract program has been integrated into WebSphere MQ for z/OS Version 6.0 and you can now use multiple TCP/IP stacks.

WebSphere MQ for z/OS can now be tuned much more dynamically to respond in a flexible way to changes in application workloads. System resources such as buffers and page sets can be added and removed without needing to restart the queue manager or channel initiator.

Standards

Java Message Service (JMS) V1.1

The current version of the JMS standard is supported. The older level of interface is also included for backward compatibility.

Internet Protocol V6 (IPv6)

WebSphere MQ now fully supports IPv6 networks. IPv6 is the latest version of the IP protocol and provides many more IP addresses than are allowed under IPv4.

ESB (Enterprise Service Bus)

IBM WebSphere MQ V6.0 delivers improved ease of use and manageability to provide a flexible proven foundation for your enterprise service bus.

What is an enterprise service bus?

IBM defines the ESB as “an architectural pattern that offers a comprehensive, flexible, and consistent approach to integration”. It delivers a connectivity infrastructure that enables applications – running on different platforms, written in different programming languages, or using different programming models – to communicate with each other, with minimal if any change to the applications themselves. It is a pattern of middleware that unifies and connects services, applications, and resources within a business. Put another way, it is the framework within which the capabilities of a business’s applications are made available for reuse by other applications throughout the organization and beyond. The ESB is not a new software product – it’s a new way of looking at how to integrate applications, coordinate resources, and manipulate information. Unlike many previous approaches for connecting distributed applications, for example RPC or distributed objects, the ESB pattern enables the connection of software running in parallel on different platforms, written in different programming languages, and using different programming models.

Key characteristics of an enterprise service bus

An enterprise service bus:

- Is standards-based.
- Can enable all parts of a business to react instantly to new information.
- Minimizes risk by using industry-standard interfaces and protocols.
- Overcomes differences in platform, software architecture, and network protocols.
- Assures delivery of transactions, even when systems and networks go offline.
- Re-routes, logs, and enriches information without rewriting applications.
- Provides an infrastructure that is highly distributed and yet can be managed centrally.
- Can distribute data throughout your business and beyond to your customers and business partners.
- Spans different operating systems, programming models, application types, and locations.
- Can be deployed incrementally, project by project, to better manage expense.
- May combine new and existing technologies and standards.

Founded on enterprise application integration

One way to dramatically cut the cost of integrating business applications is to use Message-Oriented Middleware (MOM). This enables applications to package their business information as messages, which are handled by the message-oriented middleware.

A combination of WebSphere MQ servers, WebSphere Application Server deployments, and WebSphere Business Integration Message Brokers can be used to deploy an ESB. This combined network of servers can then seamlessly exchange information across your enterprise, regardless of whether the original communication began as a JMS message from WebSphere Application Server or WebSphere MQ, or as a native message from WebSphere MQ. WebSphere MQ takes care of distributing messages to where they are needed, thus removing the need to write complex networking code, with all its inherent costs. Brokering software such as IBM WebSphere Business Integration Message Broker can further enhance the flow and distribution of information by enabling the transformation and intelligent routing of messages without the need to change either the applications that are generating the messages or the applications that are consuming them. This style of connectivity is called a message-oriented architecture and it is based around applications communicating using messages.

Other notable changes

Programming for .NET and Java

Classes for Microsoft's .NET framework that enable use of WebSphere MQ functions were originally shipped as part of maintenance updates to WebSphere MQ V5.3. They are now part of the base function included in WebSphere MQ V6 and have been updated to expose the V6 MQI options. One new element of the .NET support is provision of a monitor program. This acts as a trigger monitor, which can create a .NET object in the correct transactional context when a message arrives on an inbound queue.

The Java interfaces have also been updated. The base MQI classes support the V6 MQI options. Both the base and the JMS classes can now use the client channel definition table for getting a connection to a queue manager. This can make it easier for application programmers, because the connection definition is now provided by a system administrator instead of being written in the application code. JMS-administered objects can also point to the client channel definition table, removing a potential duplication of entries with its associated likelihood of errors.

Clustering

Clustering enables the physical distribution of the providers by using the same queue name on clustered queue managers. WebSphere MQ is then responsible for handling the routing and load balancing of messages to instances of the queue. By default, the round-robin algorithm is used for load balancing.

The workload distribution algorithms used in WebSphere MQ clustering have been extended for more varied configurations. Applications running on unequally powered machines can automatically be sent a correspondingly unequal workload.

System designers can implement queue manager topologies that redirect work around a cluster in more complex ways than a simple round-robin distribution. No additional user-written exit code is necessary.

Shared queues on z/OS

This release completes the shared queue function that has been made available in stages across multiple WebSphere MQ for z/OS releases. The restriction that messages on shared queues had to be less than 63KB has now been removed. Application programmers can use shared queues in the same way that they use private queues. The same 100MB limit applies to both types of queue.

The implementation of 'large messages', which are considered to be anything above 63KB, puts the body of the message in a shared DB2 table, with only control information and a pointer to the message being stored in the Coupling Facility (CF). The CF has comparatively limited storage space, and so this design keeps large amounts of data outside the CF. Messages that are under the 63KB barrier are stored in the CF exactly as in previous releases of WebSphere MQ. There are performance considerations with the use of DB2, which might make shared queues less desirable for applications where most or all of the messages are large. However, designers might prefer to trade any performance degradation for the improved availability that shared queues provide.

Web services

A component is included in WebSphere MQ V6 that makes it possible to run Web services over a WebSphere MQ transport layer instead of the more common HTTP protocol. The SOAP contents of a Web service request can then be carried over a reliable protocol, before being presented to a server program. Using WebSphere MQ as the transport is completely transparent to the application code. Defining an appropriate URI when deploying the service components will select the WebSphere MQ transport. Support is provided for both Axis and .NET environments, and it will also interoperate with the new CICS Web services gateway.

This also provides a basis for future work around asynchronous Web services. As standards are developed and clarified, the use of WebSphere MQ as a common asynchronous transport is expected to grow.

Additional new functions

Additional new functions include:

- JMS clients can use the channel definition table and run the same channel exits as other clients.
- The security interfaces used by **setmqaut** are given PCF equivalents to permit remote administration.
- Some error log entries can be suppressed or duplicates ignored.
- Queue managers on all platforms can automatically start items such as applications, listeners, and trigger monitors.
- Use of GSKit on Windows brings cross-platform consistency for SSL configuration.
- Support for Web services through the integration of SOAP SupportPac MA0R.
- Support for publish/subscribe through the integration of SupportPac MA0C.

- Support for the IBM Message Service Client for C and C++ (through SupportPac IA94).

Functions no longer supported

The following functions, provided in previous levels of WebSphere MQ, are not supported in WebSphere MQ V6.0:

- 32-bit hardware support for AIX, HP-UX, and Solaris; 32-bit application support remains.
- Distributed Computing Environment (DCE); any applications that use the '*_d' libraries on HP-UX must be relinked.
- UDP transport protocol on AIX.
- The Windows NT platform.
- Microsoft Management Console (MMC) snap-ins on Windows (replaced by Eclipse plug-ins).
- Use of MMC to control automatic start of programs on Windows (replaced by queue manager-controlled capabilities).

CONCLUSION

WebSphere MQ is IBM's award-winning middleware for commercial messaging and queueing. It is used by thousands of customers in every major industry in many countries around the world. In this article we have discussed the major enhancement in WebSphere MQ Version 6. Many customer requirements have been included to enable easier application development and system management. This article discusses in detail the new features and functions which make WebSphere MQ V6 a successful product.

T S Laxminarayan (ts_laxminarayan@yahoo.com)
Systems Programmer (India)

WebSphere Portal installation on z/Linux – part 2

This month we continue the article describing the multi-tier installation of IBM WebSphere Portal Server with various bundled components on a z/Linux platform.

IBM HTTP SERVER INSTALLATION

The IBM HTTP Server is an Apache-based Web server modified by IBM to provide proprietary modules. The Web server can be local to the WebSphere Application Server, ie installed on the same machine as the WebSphere Portal, or it can be installed on a remote machine with the WebSphere plug-in. In this three-tier environment we installed a remote Web server with the WebSphere plug-in on an IHSNODE machine.

The plug-in file generated at the WPSNODE machine should be later copied on this IHSNODE machine.

Do the following to install the software on the IHSNODE machine:

1 Log in on SuSE host IHSNODE:

```
# ssh -X IHSNODE
```

2 Mount the shared NFS disk on the SuSE Linux:

```
# mount -t nfs NFSNODE:/opt/setup /mnt
```

3 Change to the following directory and execute the install command:

```
# cd /mnt/WPS/was/zlinux/linuxs390/WAS50
# ./install.sh
```

4 You will get the following and will be returned to the command prompt:

```
InstallShield Wizard
Initializing InstallShield Wizard...
```

```
Searching for Java(tm) Virtual Machine ...
.........
```

5 Wait for the installation wizard to appear.

6 Click English (default) for the *Language*, click *Next* and *Accept Licence Agreement*, click *Next* and choose *Custom install*.

7 Deselect all the options selected by default. Select only the Web server and the plug-in for the installed Web server. In our environment select the following components:

- IBM HTTP Server Version 1.3.26.
- Web Server Plugins:
  - IBM HTTP Server.

8 Accept default paths for WebSphere Application Server and HTTP Server installations and click *Next*.

9 You will be provided with a list of features that you have selected to install; click *Next*.

10 File copying will start. After the installation is completed, click on the *Finish* button. You will be returned to the command prompt.

11 Umount the mount point on SuSE Linux:

```
# umount /mnt
```

Installation verification

1 Ensure that the Web server configuration file, ie */opt/IBMHTTPServer/conf/httpd.conf*, has:

- The hostname listed in the 'ServerName' directive.
- The plug-in inserted correctly.

Locate the following lines at the end of the *httpd.conf* file:

```
LoadModule ibm_app_server_http_module /opt/WebSphere/AppServer/bin/
mod_ibm_app_server_http.so
WebSpherePluginConfig /opt/WebSphere/AppServer/config/cells/plugin-
cfg.xml
```

2 Start the HTTP Server:

```
# ./opt/IBMHttpServer/bin/apachectl start
```

Check the message:

```
./apachectl start: httpd started
# ps -ef | grep http
```

Check whether any HTTP processes are running.

3 Open a browser and ensure that the proxy setting is off in the browser.

4 Type the URL http://hostname. You should see the *Welcome to IBM HTTP Server* page.

To stop the HTTP Server:

```
# ./opt/IBMHttpServer/bin/apachectl stop
# ps -ef | grep http
```

Ensure that no HTTP process(s) are running.

Regenerate the plug-in

1 Ensure that the WAS machine on which you are about to generate the plug-in is still running.

2 Open the browser and type the URL http://hostname:9090/admin/ (where *hostname* is the hostname of the WAS machine where you would be generating the plug-in).

3 When prompted for the user id, enter ‘admin’ and click on *OK* (you may enter any user id as long as security for WebSphere is not enabled).

4 On the admin console, locate the *Environment* settings in the left frame.

5 Expand the *Environment*.

6 Locate and click on the *Update Web Server Plugin*.

7 Notice in the right frame the first line *Update web server plugin configuration*.

8 Click on the *OK* button in the right frame and keep observing that frame.

9 You should see:

```
Message(s)
The web server plugin configuration was updated successfully.
```

10 You have successfully re-generated the plug-in.

11 Log out from the WAS console by clicking *Logout*.

12 Close the browser.

Copy plug-in to remote Web server

On the Web server machine, because you have installed the Web Server plug-ins, you will have the same directory structure as in the WAS machine. Copy the plug-in file from the WAS machine to the HTTP Server in the same directory.

The plug-in file is located in the following path in both machines:

```
/opt/WebSphere/AppServer/config/cells/plugin-cfg.xml
```

1 On the WebSphere Application Server machine, do the following:

```
# cd /opt/WebSphere/AppServer/config/cells
# ls -l
```

The ownership and permissions will be:

```
-rw-r—r—       root       root             plugin-cfg.xml
```

2 Copy the *plugin-cfg.xml* file from the WAS to HTTP Server.

Type:

```
# scp plugin-cfg.xml <hostname>:/opt/WebSphere/AppServer/config/cells/
plugin-cfg.xml
```

where <*hostname*> is the Web server where you want to copy the plug-in.

You will be prompted for authentication information. Type *yes* at the prompt. Type the password for the Web server machine you are connecting to and copy the file to the Web server.

3 Ensure that the file is copied to the Web server.

On the Web server, perform the following steps:

```
# cd /opt/WebSphere/AppServer/config/cells
# ls -l
```

You will see the plug-in file with the same file size, ownership, and permissions as on the WAS machine:

```
-rw-r—r—        root        root              plugin-cfg.xml
```

4 Start the HTTP Server to see whether the Web server starts with the plug-in installed:

```
# ./opt/IBMHTTPServer/bin/apachectl start
```

Check the message:

```
./apachectl start: httpd started
# Type ps -ef | grep http
```

You will see some HTTP processes if the HTTP Server has started.

WEBSPHERE PORTAL SERVER CONFIGURATION

This section describes the post-installation configuration tasks for IBM WebSphere Portal Server V5.0.2.2. These steps should be performed only after the installation activities in the sections above are completed.

Remote Web server

As we will be using the remote Web server IHSNODE in our environment, we will need to configure the WebSphere Portal to use the remote Web server.

The WebSphere plug-in installed in the Web server that is located on a separate (remote) server uses an XML configuration file (plugin-cfg.xml) containing settings that describe how to handle and pass on requests to the WebSphere

Application Server made accessible through the plug-in.

The plug-in is installed on the remote Web server machine and we will have to configure WebSphere Portal for the remote Web server. Perform the following steps:

1 Locate the file */opt/WebSphere/PortalServer/config/wpconfig.properties* and create a back-up copy before changing any values:

```
# cd /opt/WebSphere/PortalServer/config/
# cp wpconfig.properties wpconfig.properties.bak
```

2 Use a text editor to open the *opt/WebSphere/PortalServer/config/wpconfig.properties* file and uncomment and verify the following properties for your environment:

Section of properties file *WebSphere Application Server properties:*

- WpsHostName=IHSNODE.company.com – this is the fully-qualified host name of the machine that is serving the WebSphere Portal pages. If your Web server is installed on the same machine with WebSphere Portal, this value is the host name of the WebSphere Portal machine. If you are using a remote Web server, this value is the host name of the machine where you installed the Web server.

  Default Value: localhost.

- WpsHostPort=80 –this is the port number that your external Web server is currently using to listen for HTTP traffic.

  Default Value: 9081.

Note:

- Do not change any settings other than those specified in these steps.
- Use / instead of \ for all platforms.

3 Save the file.

4 Enter the following command to configure WebSphere Portal for the remote Web server:

```
# ./WPSconfig.sh httpserver-config
```

5 Copy the */opt/WebSphere/AppServer/config/cells/plugin-cfg.xml* file from the WebSphere Portal machine to the remote Web server. Ensure that you copy the file to the same directory on the remote machine.

6 Restart the Web server.

```
# ./opt/IBMHTTPServer/bin/apachectl restart
```

7 Restart WebSphere Application Server and WebSphere Portal by entering the following commands from the */opt/WebSphere/AppServer/bin* directory:

```
# ./ stopServer.sh server1
# ./ stopServer.sh WebSphere_Portal
# ./ startServer.sh server1
# ./ startServer.sh WebSphere_Portal
```

8 Verify that WebSphere Portal can be accessed from the remote Web server. For example, open a browser and access the URL http://WPSNODE.company.com/wps/portal.

Use DB2 UDB as the Portal repository

By default, WebSphere Portal installs and uses a Cloudscape database to store information about user identities, credentials, and permissions for accessing portal resources. Cloudscape is a built-in Java database that is well suited to basic portal environments. However, if the demands of the portal environment require database software with greater capability and scalability, you can also configure WebSphere Portal to use a more robust database, such as DB2.

Note: it is not advisable to transfer the database after extensively using WebSphere Portal.

Creating remote databases

If you use a remote DB2 server, you must manually create the databases that are required by WebSphere Portal. Follow these steps to create the remote DB2:

1. Log in to the DB2 server machine as a user with sufficient database privileges:

```
    # ssh -X DB2NODE
    # su - db2admin
```

2. On the DB2 server machine, enter the following commands for the WebSphere Portal database:

```
    $ db2 create database wps50 using codeset UTF-8 territory us
    $ db2 update database configuration for wps50 using applheapsz
16384 app_ctl_heap_sz 8192 stmtheap 60000 locklist 400 indexrec RESTART
logfilsiz 1000 logprimary 12 logsecond 10
    $ db2set DB2_RR_TO_RS=yes
```

3. On the DB2 server machine, enter the following commands for the content publishing and feedback databases:

```
    $ db2 create database wpcp50 using codeset UTF-8 territory us
collate using identity
    $ db2 create database fdbk50 using codeset UTF-8 territory us
collate using identity
    $ db2 update database configuration for wpcp50 using applheapsz
4096 logfilsiz 4096 logprimary 4 logsecond 25
    $ db2 update database configuration for fdbk50 using applheapsz
4096 logfilsiz 4096 logprimary 4 logsecond 25
```

4. After you have created and configured the remote content publishing databases in the previous step, perform the following database commands on the DB2 server machine:

```
    $ db2 connect to wpcp50;
    $ db2 create bufferpool wpcpBP8K size 8192 pagesize 8K;
    $ db2 connect reset;
    $ db2 connect to wpcp50;
    $ db2 create system temporary tablespace wpcpTS8K pagesize 8K
managed by system using('wpcpTS8K') bufferpool wpcpBP8K;
    $ db2 connect reset;
```

5. On the DB2 client machine, set DB2COMM to TCP/IP by using the **db2set** command, as follows:

```
    $ db2set DB2COMM=TCPIP
```

6 On the DB2 client machine, **catalog** the WebSphere Portal and content publishing databases, as follows:

```
$ db2 catalog db wps50 at node dbserver
$ db2 catalog db wpcp50 at node dbserver
$ db2 catalog db fdbk50 at node dbserver
```

where *wps50*, *wpcp50*, and *fdbk50* are databases created on the DB2 server, and *dbserver* is the node name that was created after installing DB2 client on the Portal machine.

7 Log out by entering the following command:

```
$ exit
```

8 On the DB2 client machine, test your remote connection by issuing the following commands in the DB2 command window:

```
$ db2 connect to wps50 user db2admin using <password>
$ db2 connect to wpcp50 user db2admin using <password>
$ db2 connect to fdbk50 user db2admin using <password>
```

9 Restart your database server.

Configuring WebSphere Portal for DB2

After installing and setting up databases, you must configure WebSphere Portal to populate and use these databases. Follow these steps to configure WebSphere Portal to use the WebSphere Portal and member manager databases on DB2.

1 In the command prompt, change to the directory as below:

```
# cd /opt/WebSphere/PortalServer/config
```

2 Export the current database data by entering the following command:

```
./WPSconfig.sh database-transfer-export-linux
```

3 Locate the */opt/WebSphere/PortalServer/config/wpconfig.properties* file and create a back-up copy before changing any values:

```
# cp wpconfig.properties wpconfig.properties.b4db
```

4 Use a text editor to open the *wpconfig.properties* file and enter the values that are appropriate for your environment.

Note the following information:

- Do not change any settings other than those that are specified in these steps.
- Use / instead of \ for all platforms.
- There might be additional database properties other than those listed here. Only change the properties within this table; skip all other properties.

Section of properties file *Database Properties:*

- *DbSafeMode=false* – this applies only to database-specific tasks. If this property is set to *true* no database-specific updates will be performed, such as create/ modify/drop database. The property should be used if a preconfigured database (from a previous installation) already exists. If the property is set to *false* the database will be updated, and the pre-existing database configuration is overwritten.

  Default value: false.

- *DbType=db2* – the type of database that is used to store information for WebSphere Portal. This value indicates the type of database you are using.

  Default value: Cloudscape.

- *WpsDbName=wps50* – this value represents the database name (or alias name if remote) where you want the WebSphere Portal objects created. Note: this value is also the database element in the DbUrl property.

  Default value: wps50. Note: this value cannot exceed eight characters and can contain only letters and numbers. Refer to DB2 documentation for more information.

- *DbDriver=COM.ibm.db2.jdbc.app.DB2Driver* – this is the name of the JDBC provider that is used to import SQL files.

  Default value: com.ibm.db2j.jdbc.DB2jDriver (Cloudscape).

- *DbDriverDs=COM.ibm.db2.jdbc.DB2Connection PoolDataSource* – the data source for the JDBC provider that WebSphere Portal uses to communicate with its databases.

  Default value: com.ibm.db2j.jdbc.DB2jConnection PoolDataSource (Cloudscape).

- *DbUrl=jdbc:db2:wps50* – the database URL that is used to access the WebSphere Portal database with JDBC, where <*hostname*> is the name of the remote server and <*port*> is the port where the appropriate database instance is listening. The value must conform to standard JDBC URL syntax. Note: the database element of this value should match the value of WpsDbName.

  Default value: jdbc:db2j:wps50;create=true (Cloudscape).

- *DbUser=db2admin* – this value should be an administrative user in the database. The administrator is necessary only if WebSphere Portal is creating the database. A user is necessary to connect to WpsDbName.

  Default value: db2admin.

- *DbPassword=<password>* – this password must match the password for the database user ID that is indicated in DbUser.

  Default value: password.

- *DbLibrary=C:/Program Files/SQLLIB/java/db2java.zip* – DbLibrary is machine specific. You must locate the *db2java.zip* file on your machine. For example, *C:/Program Files/SQLLIB/db2java.zip*, *C:/Program Files/SQLLIB/java12/db2java.zip*, or */home/db2admin/sqllib/java/db2java.zip*. Note: in Unix environments, you must use a colon (:) instead of a semicolon (;) to separate multiple paths. Note: make sure that you type the path to the correct Java library if you have more than one on your machine.

Default value: *wp_root/shared/app/cloudscape/db2j.jar*, where *wp_root* is the WebSphere Portal installation directory (Cloudscape).

Editor's note: this article will be concluded next issue.

Hozefa Rupawala (UAE)

WebSphere and WebSphere Application Server 6.0

WEBSPHERE APPLICATION SERVER 6.0: OVERVIEW AND NEW FEATURES

As the foundation of the WebSphere software platform, WebSphere Application Server V6.0 is the industry's premier Java-based application platform, integrating enterprise data and transactions for the dynamic e-business world. Each configuration available delivers a rich application deployment environment with application services that provide enhanced capabilities for transaction management, as well as the security, performance, availability, connectivity, and scalability expected from the WebSphere family of products.

WebSphere Application Server, built on open standards, helps you deploy and manage applications ranging from simple Web sites to powerful e-business solutions. It is J2EE-compliant and provides a portable Web deployment platform for Java components, XML, and Web services that can interact with databases and provide dynamic Web content. Included with this release is the Web services specification, WS-I Basic Profile 1.1, J2EE (Java 2 Enterprise Edition) 1.4, as well as the Java API for building Web services and clients using Remote Procedure Calls (RPC) and XML, and JAX-RPC.

Also introduced are several other areas of improvement, such as new packaging and installation of the product, using industry standards, advanced security-rich infrastructure, simplification of deployment and administration practice, and improvement in enterprise class deployment – all delivered to enhance reliability, scalability and interoperability.

My intention in this article is to help shed light with respect to the new functions and features provided with WebSphere 6, and to show how it surpasses previous releases of WebSphere Application Server. This article provides in detail what is new and changed in Version 6. The audience is anyone who is evaluating or planning to use Version 6, system administrators, programmers and particularly users of previous versions or editions.

WEBSPHERE APPLICATION SERVER FAMILY: A BRIEF INTRODUCTION

IBM's WebSphere Application Server family enables businesses of all sizes to connect the right people with the right information to increase business responsiveness and relieve business process pain. By delivering strong capabilities for integrating and streamlining business processes, WebSphere Application Server enables users to gain insight from business information across and beyond the enterprise. Moreover, the WebSphere Application Server family provides the infrastructure foundation that enables the integration of business components and processes. This foundation allows companies to innovate while leveraging legacy IT investments across heterogeneous environments.

The entry point for users is WebSphere Application Server – Express, an easy and affordable entry point for e-business that supports the complete J2EE 1.4 compatible programming model – including servlets, JSPs, EJBs, and Web services. WebSphere Application Server – Express provides an out-of-the-box solution that facilitates the construction and deployment of static and dynamic Web pages and Web applications. WebSphere Application Server, the core offering of the family, is an application platform for scalable e-business applications. WebSphere Application Server can be deployed as a single server or through a network deployment configuration that supports manageable clustering in highly-available distributed environments. WebSphere Application Server for z/OS was designed to deliver the full capabilities of the WebSphere Application Server in an environment tuned to take advantage of the unique features of z/OS and OS/390, including the ability to deliver near-zero downtime support for critical business applications, tight transactional integration with existing mainframe resources, and the ability to scale in high-availability environments built with parallel sysplex clustering. Finally, IBM offers WebSphere Extended Deployment, which 'extends' WebSphere Application Server Network Deployment to deliver a high-performance, manageable, and dynamically-scalable environment for distributed WebSphere applications.

FOCUS AREAS: AT A GLANCE

The main new items in installation, migration, coexistence, and interoperability are as follows:

- Selective support for specific 64-bit platforms.
- Selective support for Internet Protocol Version 6 (IPv6), the next evolution in Internet Protocol.
- Simplified operating system set-up and prerequisites.
- Updated launch pad application for planning and starting installation more easily.

- More componentized installation procedures let you install only what you need.
- Smaller disk image footprint.
- Binary files are shared by installations on the same machine.
- Updated First Steps application for getting started quickly after installation.
- Simplified service.
- Easier removal and reinstallation.
- Better indicators of installation status.
- Silent installation requires licence acceptance.
- New migration wizard.

The biggest changes in system administration, monitoring, and tuning can be summarized as follows:

- Changes to the default configuration.
- Fine grained application update and other deployment improvements.
- Enhanced administrative infrastructure, with J2EE 1.4-related changes.
- Growing set of administrative commands.
- Improved installation and configuration, with profiles.
- Improved monitoring and performance tuning.
- Default messaging provider.
- Even more administrative improvements!

The main improvements in the documentation, help, and tutorial are as follows:

- New ‘How do I?’ feature gathers documentation, multimedia demonstrations, tutorials, cheat sheets, and presentations for commonly performed tasks.

- Improved installation documentation.
- Improved scripting documentation.
- Conceptual overviews and 'what is new' topics have better content and are easier to find.
- Quick start shortcuts direct you quickly to key materials.
- New *Learn about WebSphere* applications section for studying the programming model.
- Improved task-based table of contents.
- Improvements to troubleshooting content and placement.

AREAS OF IMPROVEMENT: WHAT'S NEW IN THIS RELEASE

High Availability services

WebSphere Application Server V6.0 can be used as part of an overall 99.999% availability solution.

New to WebSphere Application Server 6 is a built-in High Availability Manager (HA Manager). In the event of a power failure or network outage, for example, WebSphere 6 can detect the problem and transfer the affected server's workload to another available resource. In addition, WebSphere checks its transaction history and hands off a log of what the server was doing when it suddenly became unavailable, so that any in-progress Web transactions can be resumed and completed in seconds as opposed to minutes.

The manager works with a number of servers. This is a feature of the WebSphere Application Server Network Deployment product and is an extension of the earlier failover and workload management feature. The manager monitors the health of the different servers, noting whether they become unresponsive in any of several different ways. When it determines that one or more servers (it could be an entire WebSphere cluster) is no longer available, it can shift the work that was intended for those servers to somewhere else in that data centre or perhaps to another data centre far away. Then it goes in and it grabs the transaction log, assuming it is available in shared storage, and hands it off for completion of in-flight work. So the new features are about greater flexibility that yield better availability, and these translate to real business benefits.

Service-oriented architecture building block

With WebSphere 6, IBM is also bolstering its arsenal of tools for companies that want to build Service-Oriented Architectures (SOA). Key to any SOA platform is a standards-based messaging infrastructure. Individual services within an SOA often need to be able to communicate asynchronously. WebSphere Application Server Version 6 meets this requirement with a new Java messaging engine that fully implements the Java Message Service (JMS) 1.1 specification. The new messaging engine delivers ease of use improvements. Administrative tasks such as configuring, clustering, and problem determination are now consistently performed using the WebSphere administration console. Customers can get up and running quickly because most settings and tuning parameters are set by default. The new messaging engine also delivers performance, fail over, and scalability improvements by leveraging the underlying WebSphere Application Server infrastructure. Finally, the Version 6 messaging infrastructure enables applications and services built on WebSphere to become 'Enterprise Service Bus-ready'. An Enterprise Service Bus (ESB) is a natural component of most SOA initiatives because it provides a standardized company-wide communications infrastructure for connecting services. If a company has an investment in IBM WebSphere MQ, it can easily establish an ESB consisting of WebSphere Application Servers, WebSphere MQ queue managers, and WebSphere Business Integration brokers.

This mixed network of servers can then communicate regardless of whether the original format of a message is JMS from MQ, JMS from the application server, or even native MQ.

An SOA delivers greater flexibility, enabling new composite applications to be built from existing applications and services. This raises a new requirement for transaction consistency across the SOA platform and this heterogeneous mix of services. Businesses require the ability to use transaction data from across the organization and value chain without having to reprogram the outcome for each application. WebSphere Application Server V6 delivers WS-Transaction to provide this value. WS-Transaction is a building block used with other specifications of Web services (eg WS-Coordination, WS-Security) to enable the coordination of actions of distributed applications and transactions, in effect guaranteeing agreed-to outcomes. This ensures transaction consistency in an SOA environment.

WebSphere Application Server V6 delivers the key Web services standards and foundation required to build a service-oriented architecture. IBM ensured that its Web services implementation complies with the WS-I Basic Profile to enable interoperability across different Web service platforms. IBM's service-oriented architecture infrastructure is geared to enhancing business flexibility with new composite applications that leverage this interoperability.

Web services

Over the past few years, a large and growing library of Java components and applications has been developed and placed into service. IT requires a better and easier way to reuse and leverage these investments so that it can respond to the business and give the business the flexibility it requires. IT needs an efficient process to take these Java assets and expose them as Web services to enable them to more easily integrate into an SOA and new Web-based environments. WebSphere Application Server V6 delivers this value in the form of JAX-RPC (JSR-101) 1.1, which lets IT developers translate, extend, and reuse deployed Java assets as Web services.

Another aspect of Web services that has been a work-in-progress involves publishing and finding Web services securely. Before WebSphere Application Server V6, custom programming could secure a Web service's integration or environment. Anything less, and the Web service was at risk unless it was isolated behind multiple IT security layers and implemented as a simple point-to-point integration. As the opportunity to extend these IT assets to new applications and business processes beckoned, IT discovered the need for a more robust security framework. WebSphere Application Server V6 delivers UDDI V3, enabling users to describe and discover Web services in a more secure manner. Thus, from a security standpoint, IBM has added support for the latest Web services standards to WebSphere 6 – including J2EE 1.4, the Web Services Interoperability Organization's Basic Profile Version 1.1, WS-Security from OASIS, and Universal Description, Discovery, and Integration (UDDI) Version 3. UDDI Version 3, in particular, is significant. Some key attributes of the specification are: support for digital signatures so users can verify the source of an available Web service; subscription features that let Web services consumers opt to be notified when certain services get updated; and promotion capabilities that let users take services from a departmental registry and promote them company wide.

In WebSphere 6 upgrades, the Web services standards are basically updated with point level releases. Below, you'll see some of the key features/support added with these releases:

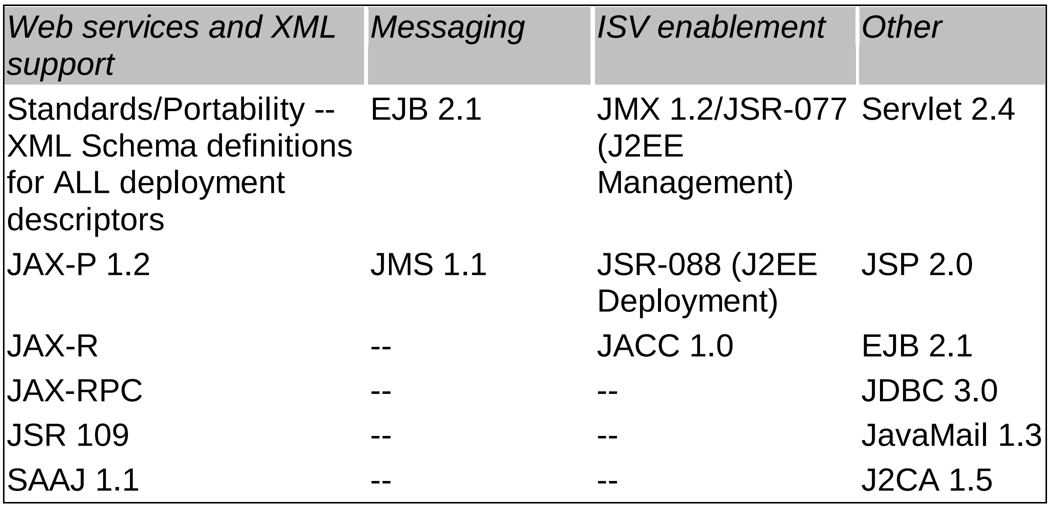
- JAX-RPC (JSR-101) 1.1 – fault support, name collision rules, new APIs, additional type support.
- JSR 109-WSEE 1.1 – utilizes J2EE 1.4 schema types, handles support for EJBs.
- SAAJ 1.2 – additional APIs for manipulating SOAP XML messages, SAAJ infrastructure now extends to DOM.

- WS-Security – applying WS-I Security Profile, OASIS draft 17.
- WS-I Basic Profile 1.1 – attachment support.
- JAXR support – access capability to UDDI registry.
- UDDI V3 support – includes registry implementation and client API library.

Some additional features that are included within the Web services support provided by WebSphere 6 are HTTP 1.1 Client support, for the use of keep-alives and zips to improve performance, JAX-RPC multi-protocol support, giving the ability to bind with RMI (Receiving Module Identification)/IIOP (Internet Inter-Operable Protocol), Extended SOAP (Small Operator Assist Program) support, and improved system management.

J2EE 1.4

WebSphere 6 supports the J2EE 1.4 programming model and extensions including servlets, JSPs (Java Server Pages), EJBs (Enterprise Java Beans), and Web services. Included



Web services and XML support	*Messaging*	*ISV enablement*	*Other*
Standards/Portability -- XML Schema definitions for ALL deployment descriptors	EJB 2.1	JMX 1.2/JSR-077 (J2EE Management)	Servlet 2.4
JAX-P 1.2	JMS 1.1	JSR-088 (J2EE Deployment)	JSP 2.0
JAX-R	--	JACC 1.0	EJB 2.1
JAX-RPC	--	--	JDBC 3.0
JSR 109	--	--	JavaMail 1.3
SAAJ 1.1	--	--	J2CA 1.5

Figure 1: Standards and point releases

are additional programming model enhancements that provide a secure foundation for an SOA.

With WebSphere 6 you can now leverage SDK 1.4.2 and new features such as SDO (Service Data Objects) and JSF (Java Server Faces). Figure 1 provides you with the standards and point releases that are a part of WebSphere 6.

Profiles for WebSphere Application Server 6.0

WebSphere Application Server V6.0 introduces the concept of Server Profiles.

In Version 5.0/5.1 there was the ability to create multiple instances of WebSphere Application Server. The **wsinstance** command creates multiple configuration instances of one initial installation of the product. The command would work for either the base WebSphere Application Server or the Deployment Manager. With Version 6 this feature is now enhanced and known as Server Profiles. Server Profiles create independent configurations of WebSphere Application Server from a set of shared libraries on the machine.

Profiles can be thought of as a specific server run-time environment operating within a separate instance of the JVM. Each run-time environment has its own configuration files, logs, properties, and other attributes. Profiles can make each Java 2 Enterprise Edition (J2EE) application server run-time unique and separate from the server binaries and from other profiles. The separation of static binaries from configuration files provides a number of benefits for system administrators.

The WAS V6.0 profiles are similar to the **wsinstance** tool provided with WAS V5.*x*, but have some important differences. The **wsinstance** tool creates configuration files for separate JVM instances, but shares other information across instances of WAS V5.*x*. In contrast, profiles draw a sharp degree of separation, so much so that each profile is separately administered with its own configuration, logs, J2EE applications, and other attributes. The separation that profiles provide allows for a unique instance of WebSphere that has not been available in previous releases of WAS.

More than one instance of the same IBM WAS product may be required on the same machine. This is often the case if the administrator desires to keep a degree of separation between applications and environments. This may be the case if you have a development group sharing the same WebSphere install instance. When using the WAS profiles, each profile runs as its own 1.4.2 JVM instance, thus providing a physical as well as a logical separation between instances of the product. This offers an advantage over multiple installs because it saves space. In addition, service updating is simpler since an administrator needs to maintain only a single set of product binaries. Profiles can also be created and deleted more easily than with whole product installs. With the separation of product binaries and profiles, it's possible for a system administrator to allow groups to have access to the profile information that defines their application server, while at the same time locking down the WAS run-time binaries.

Another useful aspect of profiles is the staging of application updates. In this scenario, you have updated an application and wish to test it on the targeted WebSphere Application Server. The problem is that an older version of an application is already running and being used. You could uninstall the application and reinstall the new update, yet that would require you to take the application offline and then reinstall if your test fails. A solution would be to create a new profile and test it there. The test profile will be operating in its own instance of WAS and will be logically separated, thus giving you a level of security and separation should your tests fail. Of course, it protects you only while running in the context of WAS, and not if the two applications are sharing the same database or any other shared resource the two applications may access.

Some of the new changes you'll see in WebSphere 6 include: the default product installation itself being a profile, support for creating new profiles, commands that support profiles, profile exchange between environments that can serve as back-up and restore, and updates to the PTF installation process to support profiles.

There are three profiles that ship with WAS 6.0. Standalone Server is your base default configuration, similar to the WAS 5 editions. Managed Node is an empty node for the initial installation versus the previous automatic creation of a WAS base server1 installation. Thirdly, Deployment Manager is a profile for creating the Deployment Manager for a cell. An added benefit with Server Profiles is you can now recreate configurations if, by chance, your configuration is corrupted.

Service Data Objects (SDO)

SDO is designed to simplify and unify the way in which applications handle data. Using SDO, application programmers can uniformly access and manipulate data from heterogeneous data sources, including relational databases, XML data sources, Web services, and enterprise information systems.

SDO, in a nutshell, supports three key concepts: Data Objects – data stored in a disconnected fashion utilizing an optimistic data store concept; Data graphs, for using metadata and remembering data history; and Data Mediation, where a mediator service is responsible for populating the graph with data objects from the data source.

SDO coupled with JSF improves the capability of Rapid Application Development for J2EE applications.

Java Server Faces (JSF)

JSF is a component-based framework that brings user interface development methods commonly used for building Java fat client user interfaces to the Java-based Web application development world.

In Version 5.0.2 of WebSphere, JSF had to be deployed along with your application. Beginning with Version 5.1.1, and continuing with Version 6, JSF is now integrated with the Application Server.

JSF support provides you with a standard user interface framework for Java Web applications, the ability to incorporate rich ‘Swing/AWT’ Widgets for your Web applications, Rapid Application Development by creating a UI from a set of reusable UI components, simplified binding of application data to the Web forms, state management across server requests, a simple model for wiring client-generated events to server-side application code, the ability to create custom/re-usable user interface components, and drag-and-drop IDE support.

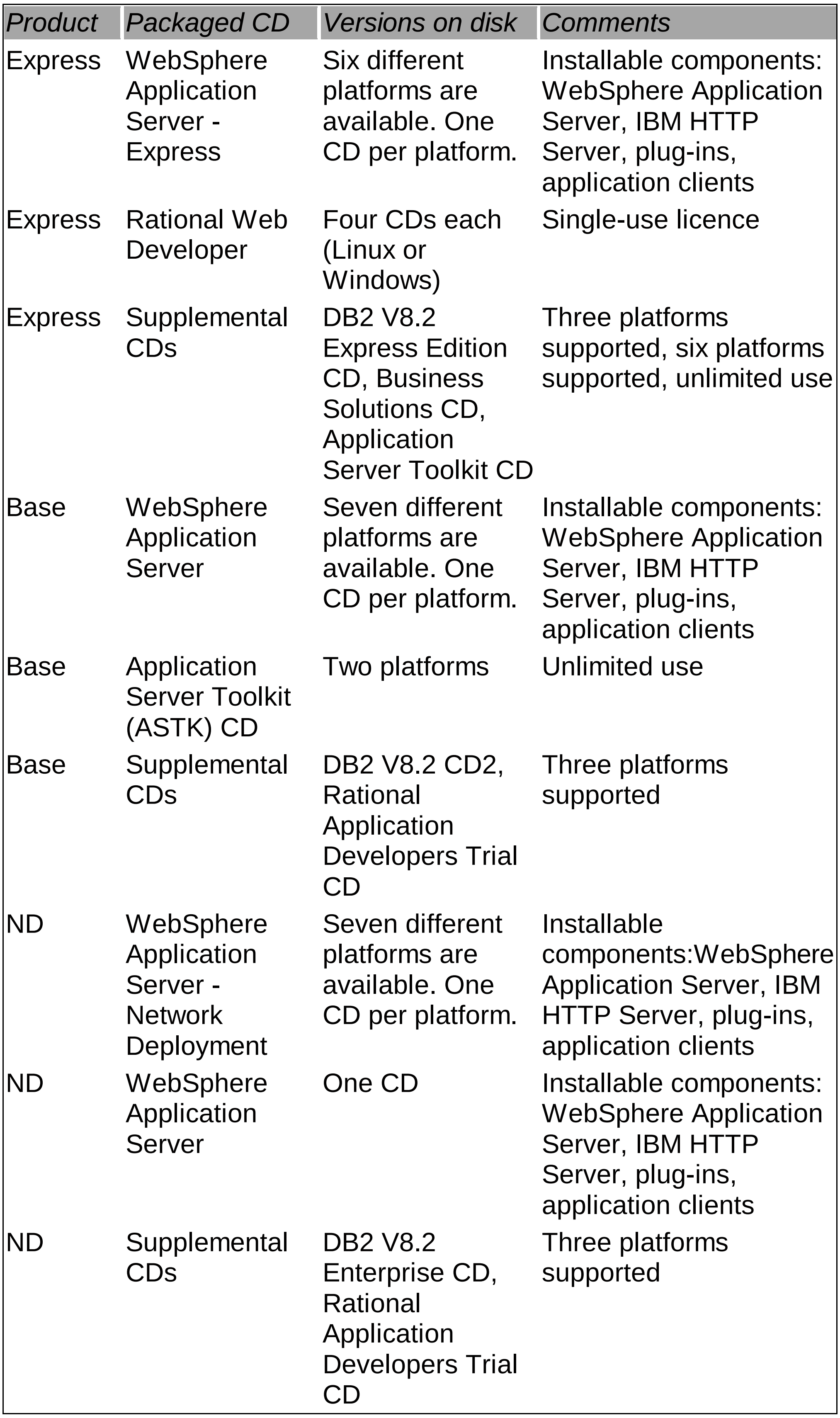
JSF brings J2EE user interface development up to a competitive level with other Rapid Application Development tooling.

Java Server Faces (JSF) also:

- Simplifies binding of application data to the UI.
- Helps manage the UI state across server requests.
- Provides a simple model for wiring client-generated events to server-side application code.
- Can be easily tooled to allow true drag-and-drop Web UI creation.
- Supports multiple rendering kits for different client devices.

New packaging and CD organization

WebSphere Application Server Version 5 and 5.1 had four flavours. The Express version was a single server version that supported JDK and a Web Container in addition to some licensing restrictions. WebSphere Application Server Base contained everything in Express plus EJB support and messaging. WebSphere Application Server Network Deployment supported clustering and multi-machines as well as edge capabilities. WebSphere Application Server Enterprise



Product	Packaged CD	Versions on disk	Comments
Express	WebSphere Application Server - Express	Six different platforms are available. One CD per platform.	Installable components: WebSphere Application Server, IBM HTTP Server, plug-ins, application clients
Express	Rational Web Developer	Four CDs each (Linux or Windows)	Single-use licence
Express	Supplemental CDs	DB2 V8.2 Express Edition CD, Business Solutions CD, Application Server Toolkit CD	Three platforms supported, six platforms supported, unlimited use
Base	WebSphere Application Server	Seven different platforms are available. One CD per platform.	Installable components: WebSphere Application Server, IBM HTTP Server, plug-ins, application clients
Base	Application Server Toolkit (ASTK) CD	Two platforms	Unlimited use
Base	Supplemental CDs	DB2 V8.2 CD2, Rational Application Developers Trial CD	Three platforms supported
ND	WebSphere Application Server - Network Deployment	Seven different platforms are available. One CD per platform.	Installable components:WebSphere Application Server, IBM HTTP Server, plug-ins, application clients
ND	WebSphere Application Server	One CD	Installable components: WebSphere Application Server, IBM HTTP Server, plug-ins, application clients
ND	Supplemental CDs	DB2 V8.2 Enterprise CD, Rational Application Developers Trial CD	Three platforms supported

Figure 2: Features by package

had async beans, application profiles, and workflow added.

You can immediately see that each version of the Application Server required customized builds. Only one had the Web Container, and another included EJB. To simplify the packaging, WebSphere 6 Express/Base Single Server CDs now deliver a single set of images that install Express or a Single Base Server and provide full J2EE 1.4.2 support for both versions.

Now included with the Express/Base product are a number of features that used to be available only in the Enterprise Edition. Most notable are asynchronous beans (now called Work Manager), and application profiles.

The CDs are now organized by platform, so now you have a single install image for Application Server. What about Deployment Manager? The Deployment Manager configuration delivers advanced deployment services that include clustering, edge-of-network services, Web services enhancements, and high availability for distributed configurations.

Which leaves what was once called WebSphere Application Server Enterprise Edition. The Enterprise Edition has been re-branded as the WebSphere Business Integrator Foundation Server. The Workflow components feature from previous editions has remained in this environment.

If you've ever downloaded an electronic version of WebSphere, I'm sure you were amazed at how many file downloads you needed to initiate. Version 5.1 had a grand total of 43 downloads. The same is true with receiving CDs. While the sheer number was daunting, the real problem was identifying which CDs were needed in order to install the product.

With WebSphere 6, you'll now have a single install image for the Application Server and Deployment manager. This greatly reduces the number of product CDs, and the CDs are organized by platform, so it's easier. Also separated are the DB2 Directory Server *et al* in a completely separate CD package. Figure 2 shows the features by package.

SECURITY FEATURES: SECURING APPLICATIONS AND THE APPLICATION SERVING ENVIRONMENT

The biggest improvement in security involves the set of supported specifications.

External JACC provider support

The Java Authorization Contract for Containers specification (JACC) Version 1.0, introduced in WebSphere Application Server Version 6.0.*x* and defined by Java 2 Platform, Enterprise Edition (J2EE) Version 1.4, defines a contract between J2EE containers and external authorization providers. Based on this specification, WebSphere Application Server enables you to plug in an external provider to make authorization decisions when you are accessing J2EE resources. When you use this feature, WebSphere Application Server supports Tivoli Access Manager as the default JACC provider.

Java 2 security manager

WebSphere Application Server Version 6.0.*x* provides you with greater control over the permissions granted to applications for manipulating non-system threads. You can permit applications to manipulate non-system threads using the *was.policy* file. However, these thread control permissions are disabled, by default.

JCA 1.5 support

WebSphere Application Server Version 6.0.*x* supports the J2EE Connector Architecture (JCA) Version 1.5 specification, which provides new features such as the inbound resource adapter. From a security perspective, WebSphere Application Server Version 6.0.*x* provides an enhanced custom principal and credential mapping programming interface and custom mapping properties at the resource reference level. The custom JAAS LoginModule, which was developed for JCA principal and credential mapping for WebSphere Application Server Version 5.*x*, continues to be supported.

SSL channel framework

The Secure Sockets Layer channel framework incorporates the new IBMJSSE2 implementation and separates the security function of Java Secure Sockets Extension (JSSE) from the network communication function.

Web authentication using the JAAS programming model

WebSphere Application Server Version 6.0.*x* enables you to use the Java Authentication and Authorization Service (JAAS) programming model to perform Web authentication in your application code. To use this function, you must create your own JAAS login configuration by cloning the WEB_INBOUND login configuration and define a cookie=true login option. After a successful login using your login configuration, the Web login session is tracked by Single Sign-On (SSO) token cookies. This option replaces the SSOAuthenticator interface, which was in WebSphere Application Server Version 4.

Web services security

WebSphere Application Server Version 6.0.*x* increases the extensibility of Web services security by providing a pluggable architecture. The implementation in WebSphere Application Server includes many of the features described in the Organization for the Advancement of Structured Information Standards (OASIS) Web Services Security Version 1 standard. As part of this standard, WebSphere Application Server supports: custom pluggable tokens that are used for signing and encryption; pluggable signing and encryption algorithms; pluggable key locators for locating a key that is used for digital signature or encryption; signing or encrypting elements in a SOAP message; and specifying the order of the signing or encryption processes.

Simplifying deployment and administration

Lowering IT projects’ time-to-value conveys competitive advantage to those who are driving an IT strategy linked to their business strategy. Faster business strategy execution and responsiveness to customer needs are a key goal. One of the key value propositions of application servers and development tools is rapid development and deployment to enable this time-to-value.

IT needs an easy-to-develop and easy-to-deploy application platform. Users need an easy-to-use application platform that enables applications to do their work, integrate with other IT assets, and present results and interaction screens in an intuitive manner. Developers gravitate to the tools that allow them to deliver these values in the fastest and easiest manner. WebSphere Application Server V6 delivers a simplified, yet robust development and deployment experience via new features, including Java Server Faces, WebSphere Rapid Deployment, and Tech Previews of Install Factory. Java Server Faces is a new J2EE presentation framework that simplifies user interface development for J2EE applications. WebSphere Rapid Deployment is a framework for automating and controlling the development and deployment of J2EE artefacts to reduce developer workload and time. Coined 'WebSphere Rapid Deployment', WebSphere 6 has improvements in annotation-based programming, deployment automation, and change-triggered processing. Tooling and toolkits that support these improvements are also shipped.

Annotation-based programming

Driven by the popularity of Xdoclet, and now standardized by JSR-175, annotations are being touted as the best way to represent support for orthogonal concerns such as transactions and persistence. Some of the features of annotation-based programming supported in WebSphere 6 are: annotation using Xdoclet, developers can create and maintain a single artefact, developers can insert metadata into source code, and it supports WebSphere's programming model (EJBs, servlets, JMX, Web services, etc).

As you can surmise, these features improve your ability to develop quickly. Now what about deployment?

Deployment automation

One of the keys to getting development and testing done sooner is the ability to deploy applications quickly. In the past, getting applications deployed to WebSphere, on occasion, took some effort. With WebSphere 6 you can automatically deploy on-the-fly while still within an editor. Some of the other deployment automation features are support for local and remote servers, and support for fine-grained application changes.

The automated deployment features certainly improve your ability to make changes and get them installed and running quickly.

Change-triggered processing

Another important step to make deployment easier is the ability to make small changes to the deployed application without a full reinstallation. Some of these improvements are supported in WebSphere 6, such as the ability to execute operations based on certain changes in the key artefacts of the application, the ability to create new artefacts based on existing artefacts, and the enablement of a ‘hot directory’ concept for quick updates and deployment. These features certainly improve ‘hot deployment’.

Application Server Toolkit

The ASTK (Application Server Toolkit) that began with Version 5 has been updated in Version 6. The ASTK provides WebSphere developers and administrators with a set of Eclipse-based tools for assembling, deploying, debugging, and profiling J2EE applications.

Let’s briefly review some of the features. Included in the Version 6 ASTK is Eclipse 3.0 workbench and JDT (Java Development Tooling). JDT allows users to write, compile, test, debug, and edit programs written in the Java programming language.

Also included are deployment descriptor editors, including the

WebSphere extensions and bindings, Module/EAR creation and manipulation, debugging and profiling/tracing tools, and Application Server Tooling with configuration validation, and local and remote WebSphere test environments.

In addition, you'll find the Universal Test Client, Web services deployment tooling, EJB deployment tooling, and EJB CMP/RDB mapping. All of these features, along with WebSphere's Rapid Deployment capabilities, really enhance the development and deployment capabilities of WebSphere 6.

Other notable changes

Also in WebSphere 6, IBM has rewritten the messaging engine inside the application server to support the latest version of the Java Message Service messaging standard. It's faster now, since the engine can run in the same Java virtual machine as other applications, eliminating the need to cross processing boundaries. The retooled messaging engine is aimed at making it easier for users to link new business applications to an existing messaging backbone, such as IBM's WebSphere MQ. IBM's application server now handles some of the configuration tasks, such as translating between protocols.

IBM also looked to streamline the way WebSphere 6 handles the myriad communication protocols it supports, such as TCP/IP, HTTP, Secure Sockets Layer, and others. It found ways to share code, so that the process of adding protocol support is more efficient. IBM created a new component of the application server called a transport channel service. It's a pluggable architecture that uses adapters to handle communications with the application server. The end result is a doubling of the number of possible simultaneous connections to a given application server.

To save time for developers who build applications with WebSphere, Version 6 includes a new wizards-based drag-and-drop environment that automates some common steps of application development and deployment. Using the wizards, developers can purportedly reduce by up to 75% the number of programming steps previously needed to build an application.

There's a lot packaged in WebSphere 6. The features you may want to consider researching deeper are administration changes, including an enhanced EAR file, the ability to introduce small changes into already installed applications, Add/Remove modules from installed applications, improved template structure, support for more templates for different machines and environments, new binary packaging, and safeguards preventing applications like the admin console or file transfer applications from being deleted.

You'll also find the ability to mix versions of WebSphere within a cell, a new administration console look and feel, with the ability to customize the console, and an integrated Web performance viewer.

In addition, Enterprise Deployment brings transport channel services, providing higher connection rates with non-blocking I/O, Web container and message improvements (such as sharing ports and thread pools), all new JMS embedded support (including persistent messaging, clustered queues, and connectivity to the MQ backbone), high-availability featuring hot standby, peer failover, enhanced data replication (improving performance and scaling), Java Web Start support, and better documentation.

SUMMARY

The IBM WebSphere Application Server 6 provides a programming model with a set of consistent services for developing transaction-oriented applications and providing the foundation for integrating other applications, data, and business processes. This article has discussed in detail the new features and functions in WebSphere Application Server 6 and how it provides you with a better environment.

T S Laxminarayan (ts_laxminarayan@yahoo.com)
System Programmer (India)

Where’s WMQ going?

Some clues to the future of WebSphere MQ can be found in recent IBM announcements.

Ascential has now been completely absorbed into IBM and ex-Ascential president Peter Fiore is now Vice President of IBM’s newly-formed Information Integration Solutions division. About 400 of his old staff are now working for IBM.

Not surprisingly, there are also some overlaps between Ascential’s DataStage TX transactional data integration product (which started life belonging to Mercator Software until Ascential bought the company in 2003) and the WebSphere Business Integration Message Broker (formerly known as MQ Integration Broker).

IBM says that it will sell both products and work to eliminate the overlaps.

IBM also gained another product from Ascential, code-named Hawk. IBM has started to build ties between Hawk and integration software it was developing under the code name Serrano.

The difference between the two companies’ products is quite major. Ascential’s software is primarily used to take data from multiple sources and compile it in a single place in order to analyse trends and generate reports. Conversely, WebSphere Information Integrator (which is what Serrano is building on) is designed to query different data sources at the same time, without consolidating information in a single place.

Hawk and Serrano are both being beta tested at the moment.

Nick Nourse
Independent Consultant (UK)

MQ news

IBM has announced a beta program for new software that allows all types of information to be more effectively used to power business intelligence systems, consolidate enterprise applications, create and manage master repositories of critical business information, and search and access information on demand no matter where it is stored within the enterprise - the company claims.

IBM is combining technology from its acquisition of Ascential Software (code-named Hawk) with IBM's existing information integration portfolio (project Serrano).

Project Serrano streamlines the design and development of information infrastructures and enables users to leverage information across the enterprise through enterprise search capabilities, information integration, and text analytics. IBM Rational Data Architect is built on the Eclipse platform and helps data architects model, discover, map, and analyse data across multiple information sources, automating information integration in complex environments.

Also announced is WebSphere Information Analyzer (part of project Hawk), which provides end-to-end data profiling, auditing, and business-oriented analysis in a single product. It shares a central repository with IBM WebSphere DataStage and IBM WebSphere QualityStage, allowing immediate metadata sharing and traceability across those products.

For further information contact:
URL: www.ibm.com/software/data/integration/launch.html.

* * *

PolarLake has announced the availability of free Proof-Of-Concept (POC) programs for IBM customers wishing to utilize PolarLake's Enterprise Service Bus product alongside their existing IBM technology investment.

The POC program includes access to all relevant PolarLake products – including the Integration Suite – and five days of on-site professional services. During this time PolarLake can design, build, and create real applications, solving integration challenges and managing the orchestration and mediation of software services – without code. Because PolarLake works seamlessly alongside the existing IBM product stack – including WebSphere MQ, WebSphere Application Server, and DB2 UDB – IBM customers are able to deploy these applications within their own environments in order to confirm the suitability of the ESB approach to their own specific requirements.

For further information contact:
URL: www.polarlake.com/en/html/news.

* * *

Rogue Wave Software has announced Version 2.5 of its LEIF framework that features application development tools to incorporate business logic in different formats.

LEIF allows enterprises to use existing applications and systems to create new SOA applications using new or existing C++ code. The framework is compatible with message transports such as WebSphere MQ and Tibco Rendezvous, as well as HTTP, POP3, SMTP and SSL.

For further information contact:
URL: www.roguewave.com/products/leif/index.cfm.