A technical discussion of application server performance

April 2002

WebSphere® software

Using a Web application server to provide flexible and scalable e-business solutions.

By Michael Malzacher, IBM WebSphere Marketing Group and Thomas Kochie, IBM WebSphere Performance Group

Contents

Introduction

The growth of network-based applications providing e-business and enterprise services continues at a rapid pace. Web application servers have evolved as the primary platform used to develop, deploy, manage and analyze these services. Application servers provide the foundation for Web-based applications delivering the programming model and infrastructure for data-driven, dynamic and personalized Web sites. Increased use – coupled with expectations for availability and performance – places high demands on Web application server technology to meet customer requirements.

IBM WebSphere® Application Server, Version 4.0 provides flexible configurations to achieve these application requirements across a broad range of deployments. WebSphere applications can range from small intranet servers to large e-commerce sites, each having different requirements for performance, scalability and availability. This white paper provides a performance characterization of IBM WebSphere Application Server, Version 4.0 and describes new enhancements in performance and its greater performance capabilities. The results were obtained using industry-wide benchmark applications and are certified by ECperf Expert Group, an independent, third-party group representing the Java™ technology community. The data clearly demonstrates that IBM WebSphere Application Server performs at the level required by today's demanding e-business applications. WebSphere Application Server, Version 4.0 provides significant improvements over earlier releases, including improvements to the core elements of the WebSphere Enterprise Java programming model.

The next section highlights some of the technology advantages found in WebSphere Application Server, Version 4.0 and includes specific workload and price/performance comparisons. It summarizes benchmark results and provides more information about the newest WebSphere Application Server performance technology enhancements and performance tools, such as the dynamic cache option and the WebSphere Performance Benchmark Sample application. In addition, this white paper includes a section on how to estimate capacity planning, along with information on IBM High Volume Web Site Simulator and how this unique WebSphere tool can simplify capacity planning. The last section provides additional sources for accessing customer case studies and press announcements, as well as how to obtain more information about WebSphere Application Server, Version 4.0.

New performance advantages of WebSphere Application Server, Version 4.0

With its new enhancements in performance technology and performance tools, WebSphere software provides a highly competitive solution to other implementations of the open Java 2 Platform, Enterprise Edition (J2EE) programming model, as well as alternative technologies like CGI programs and Active Server Pages. As this white paper illustrates, WebSphere Application Server, Version 4.0 provides a flexible and scalable solution appropriate for a range of environments – from small departmental servers to large e-commerce deployments.

WebSphere Application Server, Version 4.0 offers the following performance technology advancements:

- *Enhanced scalability on large SMP systems. Several areas of the WebSphere server runtimes show significant improvement in reducing lock contention, which improves scalability on symmetric multiprocessing (SMP) systems.*
- *HTTP session performance. Additional performance improvements were shown for persistent sessions.*
- *Dynamic fragment cache. This WebSphere innovation enables parts of servlet and JavaServer Pages (JSP) files to be cached, greatly improving overall performance.*
- *IBM Developer Kit for the Java Platform, Version 1.3. This is the latest version of the developer kit and contains many functional and performance enhancements, including improvements in the just-in-time (JIT) compiler, garbage collection and memory management.*
- *Built-in Web server. The WebSphere built-in HTTP server can be used in conjunction with an external Web server or in a standalone configuration.*
- *JSP files and servlets performance converging. Version 4.0 offers better overall performance and flexibility because JSP files converge on the same level of performance as servlets.*
- *Object request broker (ORB) performance. WebSphere contains ORB-level improvements in marshaling complex data types such as vectors and hash tables.*

WebSphere Application Server, Version 4.0 offers the following performance tools enhancements:

- *Resource Analyzer. The Resource Analyzer is a performance monitor for WebSphere Application Server, Version 4.0 runtime resources and application resources.*
- *Performance Monitoring Instrumentation API. Performance Monitoring Instrumentation (PMI) is an API used by components of the WebSphere server to capture performance-related metrics in realtime. Metrics recorded by the PMI are reported through the Resource Analyzer, through IBM tools including Tivoli® software and other third-party tools.*
- *Performance Tuner Wizard. The Performance Tuner Wizard is a tool included in WebSphere Application Server, Version 4.0 that provides configuration support for the most common performance-related settings associated with your application server or server group. In addition, this tuner wizard is integrated with the IBM DB2® Tuning Wizard for end-to-end tuning assistance.*
- *WebSphere Performance Benchmark Sample download. The latest version of WebSphere Performance Benchmark Sample (also referred to as Trade e-business benchmark) is available on the Web and contains various improvements that produce a realistic assessment of application server performance.*

The data in this white paper was obtained by WebSphere Performance Benchmark Sample and is available at:
www-4.software.**ibm.com**/webapp/download/search.jsp?go=y&rs=benchmark

Resource Analyzer and performance monitoring

One of the enhancements found in WebSphere Application Server, Version 4.0 is the Resource Analyzer. The analyzer retrieves performance data by periodically polling the administrative server. Data is collected continuously and retrieved as needed from within the analyzer. The impact rating of data to be collected is specified by setting the instrumentation level using the WebSphere administrative console. The Resource Analyzer's graphical interface provides controls for choosing particular resources and performance data to include in a view.

The Resource Analyzer provides a wide range of performance data for two kinds of resources: application resources (enterprise beans or servlets, for example) and WebSphere runtime resources (such as thread pools, connection pools and Java VM memory). Performance data includes statistical data (like the method response time) and load data (such as average connection pool size during a specific time interval). This data can be reported for individual resources or aggregated for multiple resources.

Depending on which aspects of performance are being measured, the Resource Analyzer can manipulate data to accomplish the following tasks:

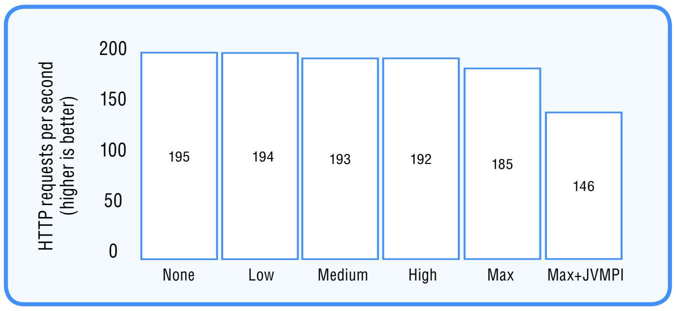
- *View data in realtime or view historical data from log files.*
- *View data in chart form, allowing comparisons of one or more statistical values for a given resource on the same chart.*
- *Record current performance data in a log and replay from a previous session.*
- *Compare data for a single resource to an aggregate of resources on a single node.*

With this data, the Resource Analyzer can be used to do the following:

- *Monitor realtime performance, such as response times for servlet requests or enterprise bean methods.*
- *Detect trends by analyzing logs of data over time.*
- *Determine the efficiency of a configuration of resources (such as the amount of allocated memory or database-connection pool sizes).*
- *Gauge the load on application servers and the average response wait-time for clients.*

Data collection and reporting incur a cost—they can affect performance of your distributed application. Although performance data is automatically collected at all times, it is not reported by default. Instead, you must explicitly enable data reporting for those aspects of your system to be monitored. The administrative console provides a dialog box for specifying which data is to be collected for which resources. When data reporting is enabled, the types of data displayed depend on user-defined instrumentation levels. These levels allow you to choose among high-impact and low-impact data, and thus control the overall performance impact of the data-collection process itself.

Resource Analyzer is a tool used to measure PMI and is designed for a production environment and to keep the data collection and reporting costs to a minimum. The results of the performance testing shown in Figure 1 demonstrate this low cost. Even with the high level of monitoring for all possible selections, the impact to overall performance was only 1.1 percent.



Maximum: Includes EJB methods level response time and access count data
Maximum+JVMPI: Includes starting Java VM with JVMPI setting and provides Java VM garbage collection, allocation and lock data

Figure 1. PMI overhead for WebSphere Performance Benchmark Sample (Enterprise JavaBeans components)

Dynamic fragment cache

One of the newest performance enhancements of WebSphere Application Server, Version 4.0 is an optional dynamic cache for servlets and JSP files. This in-memory cache uses hooks to the application server's servlet engine to intercept calls to a servlet's service() method and checks whether the request can be served from the cache. When a servlet executes, the cache builds a unique key for that invocation (from the values of specified request variables), stores its output and performs other actions, such as setting HTTP response headers and executing other servlets. To enable and configure the cache, users define cache policies that can either be written in XML or attached directly to application EAR and WAR archives; no changes need to be made to existing J2EE applications to take advantage of the dynamic fragment cache technology.

Performance Tuner Wizard

Performance Tuner Wizard is another tool included in WebSphere Application Server, Version 4.0 that provides configuration support for the most common performance-related settings associated with your application server or server group.

Performance Tuner Wizard can optimize the settings for your applications, servlets, enterprise beans and data sources for anticipated workloads.

You can use Performance Tuner Wizard to change the following settings:

- *Web container*
- *Maximum thread size*
- *ORB properties*
- *Pass-by-reference*
- *ORB thread-pool size*
- *Data source*
- *Connection-pool size*
- *Prepared-statement cache size*
- *DB2® Universal Database™*
- *Uses the DB2 SmartGuide to tune the DB2 database associated with the data source*
- *Java Virtual Machine (Java VM)*
- *Minimum and maximum heap size*

For more information about Performance Tuner Wizard and descriptions of settings, visit the WebSphere Application Server InfoCenter at:
www-4.**ibm.com**/software/webservers/appserv/infocenter.html

Benchmark results overview

WebSphere Application Server, Version 4.0 performed faster and more efficiently than our competition even when tested on a variety of benchmark standards. WebSphere came out ahead when its architecture was stressed by ECperf benchmark testing, but it also scored high on Web primitive benchmark testing, which tested individual components of the server application.

All results are certified and may be obtained by clicking on the links that follow each section.

ECperf benchmark results show WebSphere Application Server, Version 4.0 provides first-tier performance at substantial cost savings

In January 2002, the ECperf Expert Group certified the benchmark results of the IBM WebSphere Application Server, Version 4.0 running on a configuration of IBM @server pSeries™ servers and IBM DB2 Workplace. WebSphere Application Server showed an impressive performance of 10316.13 benchmark business operations per second at standard workload (10316.13 BBops @ STD) and a very impressive price metric of 27 dollars per benchmark business operations at standard workload (27 $/BBops @ STD).

In February 2002, IBM submitted new certified results running Microsoft® Windows® on an Intel® platform. Results from IBM showed a very impressive workload of 16,634 BBops @ STD as well as a low $13 BBops @ STD as the price/performance metric. IBM used half the number of application server instances (two) compared to BEA (four), which demonstrates the superior scaling of WebSphere Application Server, Version 4.0. In addition, WebSphere Application Server, Version 4.0 offers high performance at only 72 percent of the total cost. IBM WebSphere Application Server, Version 4.0 delivers high performance using less hardware.

The results of this independent study, as well as the full disclosure report, are published on the independent Java Community Web site at:
http://ecperf/theserverside.com/ecperf/index.jsp?page=results/topten_performance

Look for future benchmark results of WebSphere Application Server to be published at this Web site:
http://ecperf/theserverside.com/ecperf/

What is ECperf?

ECperf is an Enterprise JavaBeans (EJB) application that measures the scalability and performance of J2EE-compliant servers and containers. Other currently available benchmark applications – such as the IBM Performance Benchmark Application (PBA), TPC-C, TPC-D and TPC-W – are designed to stress an application server from a workload and scalability perspective (including testing the GUI, presentation or database scalability). ECperf, however, is specifically designed to test the architecture of an application server and measures performance and scalability of production-ready EJB modules (including memory management, connection pooling, passivation and activation and caching). This benchmark application is being developed under the Java Community Process, in conjunction with J2EE server vendors, including IBM.

ECperf is a benchmark application based on real-world business problems using manufacturing, supply chain management and order/inventory as the target. ECperf models businesses by using the four domains shown here:

- *Customer domain: Handles customer orders and interactions*
- *Manufacturing domain: Performs the manufacturing operations*
- *Supplier domain: Handles interactions with external suppliers*
- *Corporate domain: Acts as the headquarters for customer, parts and supplier information*

The audience for the ECperf benchmark is both the J2EE user and server vendor. ECperf can assist the J2EE user in understanding J2EE scalability and tuning. An application server vendor can use ECperf to improve and showcase product performance.

For more information on ECperf, visit http//java.sun.com/j2ee/ecperf.

The ECperf specifications and kit, along with the independent results, are available at http//ecperf.theserverside.com.

ECperf benchmark results

The results obtained by ECperf are measured in *BBops*, which is an acronym for *successful benchmark business operations*. The term *BBops/min* is the ECperf metric that denotes the average number of successful benchmark business operations per minute completed during the measurement interval. (Note that the measurement interval used in the following graphics are at standard workload.) To be specific, BBops/min is composed of the total number of business transactions completed in the customer domain, added to the total number of workorders completed in the manufacturing domain and normalized per minute or: BBops/min = (transactions + workorders)/min.

(The higher the number, the better the performance of the application server.) The term *$/BBops* is the ECperf metric that refers to the total cost of the system under test (configuration) divided by the number of BBops. The lower the number, the better because it means that the total cost of ownership of the application server and related hardware is lower.

The results of ECperf price/performance comparison were obtained using this formula.

Dynamic fragment cache benchmark analysis results

WebSphere Application Server, Version 4.0 now provides a sophisticated caching algorithm to enhance performance. The advantages of dynamic fragment caching are highlighted in the following list:

- *Dynamic fragment caching technology produced a significant throughput improvement for this benchmark.*
- *The cache-priming interval is relatively brief so throughput increases become apparent shortly into the run.*
- *The newest version of WebSphere Performance Benchmark Sample EJB component workload includes end-to-end pathlength for container processing, business logic, database and so forth. A Java Database Connectivity (JDBC) application, JSP-centric application or other application with more servlet engine processing and less EJB container or database activity shows even greater throughput improvements.*

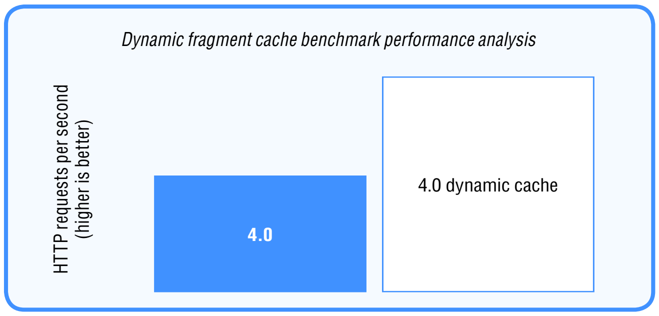


Figure 2. Dynamic fragment cache enhances the performance of WebSphere Application Server, Version 4.0.

WebSphere Performance Benchmark Sample

To evaluate the various performance aspects of WebSphere Application Server, the WebSphere performance team drew upon industry-standard benchmarks, such as TPC-W and SpecWeb99, to produce WebSphere Performance Benchmark. This benchmark is derived from experiences with a variety of customer environments and was built to emulate an online brokerage firm, which is why it is often referred to as Trade.

WebSphere Performance Benchmark Sample is a versatile test case designed to measure aspects of scalability, performance and competitiveness. It is a collection of Java classes, Java servlets, JavaServer Pages and Enterprise JavaBeans components that together form an application providing emulated brokerage services. It was developed using IBM VisualAge® for Java and IBM WebSphere Studio. Each of the components are written to open Web and Java Enterprise APIs, making the WebSphere Performance Benchmark Sample application portable across J2EE-compliant application servers.

The WebSphere Performance Benchmark Sample application allows a user, typically using a Web browser, to perform the following actions:

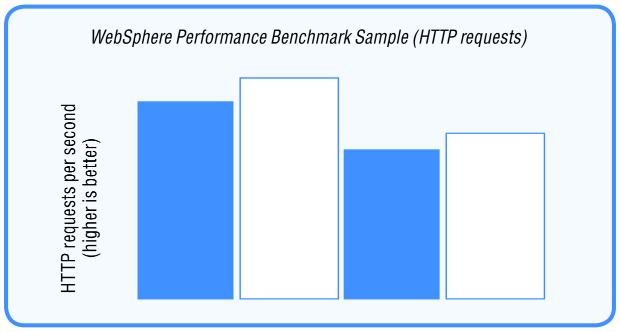
- *Register to create a user profile, user ID, password and account balance*
- *Log in to validate a previously registered user*
- *Browse current stock price for a ticker symbol*
- *Purchase shares*
- *Sell shares from holdings*
- *Browse a portfolio*
- *Log out to terminate the user's active interval*

WebSphere Performance Benchmark Sample accomplishes two things: first, it precisely measures performance characteristics of WebSphere Application Server while closely emulating behavioral patterns of real-world applications. Second, the benchmark provides a method to target and analyze individual components of WebSphere Application Server.

WebSphere Performance Benchmark Sample results

Analyzing results from WebSphere Performance Benchmark Sample and examining the performance nature of other Web application products provides the foundation for understanding general Web application performance and specific characteristics of WebSphere Application Server, Version 4.0 applications. With this data WebSphere customers and solution providers can estimate and plan for applications performance and requirements.

WebSphere Application Server, Version 4.0 outperforms WebSphere Application Server, Version 3.5.3 for WebSphere Performance Benchmark Sample JDBC components by 11 to 36 percent and WebSphere Performance Benchmark Sample EJB components by 15 to 75 percent, depending on the platform. Figure 3 shows the improved performance of WebSphere Application Server from Version 3.5 to Version 4.0. (All the benchmarking data in this report is compiled from WebSphere Performance Benchmark Sample, Version 2.531, unless otherwise noted in the data charts or analysis bullets.)



WebSphere Application Server, Version 3.5.3 ■ **WebSphere Application Server, Version 4.0** □

Figure 3. The improved performance of WebSphere Application Server, Version 4.0 compared to Version 3.5.

WebSphere Performance Benchmark Sample follows the WebSphere Application Development best practices for performance and scalability. For WebSphere best practices information, use the Performance topic pulldown menu at:
www7b.boulder.**ibm.com**/wsdd/zones/bp/

To download WebSphere Performance Benchmark Sample, visit:
www-4.**ibm.com**/software/webservers/appserv/wpbs_download.html

WebSphere Performance Benchmark Sample and Web primitive benchmarks

WebSphere Performance Benchmark Sample can be broken down into a series of simpler performance and scalability test cases for WebSphere Application Server, Version 4.0 called *Web primitive benchmarks*. These benchmark measurements are designed to test the performance of the most fundamental (or *primitive*) aspects of a dynamic Web site, including individual components of a Web application such as servlets, JSP files, HTTP sessions and EJB components.

The following are the four areas of concentration for these primitive tests that highlight characteristics of specific features or components of the configuration:

- *Component tests (such as servlet engine, Enterprise Java Service)*
- *Multiple platforms (such as Microsoft Windows 2000, IBM AIX®, Sun Solaris, Linux®, HP-UX)*
- *Multiple Web servers (such as IBM HTTP Server, WebSphere Application Server native)*
- *Multiple hardware configurations (such as UP, SMP)*

The WebSphere performance team does not exercise all possible permutations. Instead, a set of tests was selected from combinations deemed most relevant to customers. The following section discusses the Web Primitive benchmarks in detail.

PingServlet

PingServlet is the most basic ServletEngine primitive. It exercises the round-trip path of a request to execute a servlet. The followed path emanates from a Web client and travels through the Web server and Java technology-based ServletEngine. PingServlet does the very minimum of server-side processing; it simply returns a response of the string PingServlet Memory with the current date and time. By using the results of PingServlet, we can derive upper bound throughput of producing dynamic content from a Web application server.

PingSession

The HTTP protocol is stateless by design, although most Web applications need to maintain at least some state information. Cookies have emerged as the standard for maintaining state; Web Sphere Application Server, Versions 3.0, 3.5.3 and 4.0 provide a servlet session API that uses cookies to enable stateful servlets. The session data is maintained on the server, and the session ID in the cookie is used to associate the user with the session data. PingSession quantifies the additional processing required to maintain session state and counts the number of times it was invoked by the user and stores the count in the session-state data. Each time the counter reaches a preset fixed constant, a new session is created and the counter is reset.

PingJSP

The preceding servlets generate HTML responses directly within the mainstream logic of the servlet code. This model has an inherent problem because it forces your Java programmers to be HTML experts. Alternatively, it forces your expert Web site designers to learn Java.

In contrast, the JSP model lets you separate the content generation from the HTML presentation logic. In the JSP model, the servlet retrieves business data and the JSP script formats the complete HTML response. This separation provides significant advantages from a design and maintenance perspective. The performance of the JSP technology-based model is converging and catching up to the performance of servlets. Depending on specific performance requirements, JSP technology-based usage might be minimized for aggressive performance environments.

PingEntity

PingEntity tests the key function of a servlet to a session EJB component (control the transaction processes) to an entity EJB component (to actually perform a transaction). PingEntity creates a request from a browser and sends it to the HTTP server to invoke the servlet. The servlet instantiates a session bean, which in turn instantiates an entity bean. The session bean then calls certain methods to retrieve data from the entity bean. Finally, the servlet returns the request to the browser in the form of an HTML page.

PingJDBC

PingJDBC tests by using a prepared statement for database read access. Statement parameters are set dynamically on each request. PingJDBC creates a request from a browser and sends it to the HTTP server to invoke the servlet. The servlet in turn gains a reference to data source and executes a simple SQL query to the database. When this query completes, the servlet returns a response to the browser.

Web primitives testing tools and methodology

The primitive tests were exercised using a variety of hardware and software configurations, following the same methodology for each test. AKtools is a specially developed toolset used to drive the primitive benchmarks that allows a user to test a Web server, a Web site and a Web application. The two current applications are AKstress and AKrecord. AKstress is a high-performance, simple, threaded HTTP engine that simulates hundreds or even thousands of HTTP clients using a highly configurable set of directives in a human-readable and easily modified configuration file. AKrecord is a simple eavesdropping proxy that records a user's session against a Web server for later playback in AKstress. When the two applications are combined, it becomes very easy to quickly build an AKstress configuration that—with minor tuning—allows a user to evaluate the performance of Web servers, web sites or Web applications. A client load of 100 is typically used for these performance test runs. A request is defined as a single, round-trip connection between an end user and the Web server.

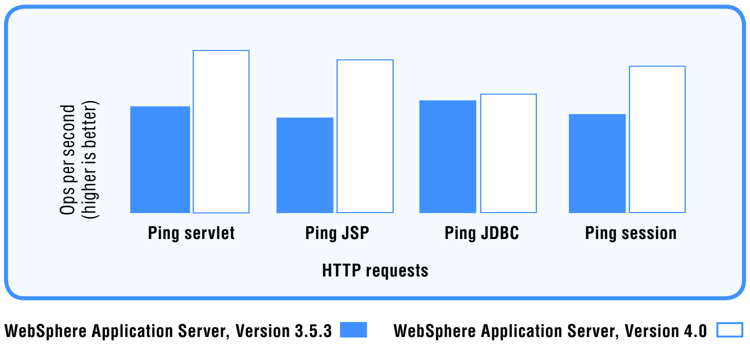


Figure 4. WebSphere Application Server, Versions 3.5.3 and 4.0 with ping servlet.

Flexible configurations for application scalability

Web sites range from small intranet servers to large e-commerce sites with vastly different requirements for performance and availability. To meet the demand across this range of requirements, an application server must not only perform well but must also scale appropriately. In addition to high performance, WebSphere Application Server, Version 4.0 provides flexible configurations to achieve application scalability across a broad range of deployments. Descriptions and performance measurements of some of these configurations are presented in the following section.

Customers are typically interested in three mainstream performance indicators:

- *Response time*
- *Throughput and the capability to sustain a large load of concurrent clients*
- *Service availability*

Approaches used to achieve the best performance for these requirements include faster processors, more processors in an SMP system and more nodes in a cluster. Each of these helps improve the performance metrics of WebSphere Application Server, Version 4.0. Clustering—supported by WebSphere Application Server, Version 4.0—helps increase client load, throughput and availability.

SMP scaling

Adding processors in an SMP system is a straightforward method for improving Web site performance. WebSphere Application Server takes advantage of multiple CPUs through its multithreaded design. The single WebSphere server process runs on a Java VM providing the infrastructure and management for the threaded system. The single process aggregates multiple threads running concurrently across SMP processors. Maximum performance benefits—low resource contention, for example—are realized when the application is specifically designed for SMP systems.

SMP scaling results

- *For WebSphere Performance Benchmark Sample JDBC components tests across all platforms, an average of a strong 3x scaling factor was realized from 1 to 4 processors on WebSphere Application Server, Version 4.0.*
- *Scalability results for the EJB modules are comparable to WebSphere Performance Benchmark Sample JDBC components scaling for the small 4-way servers.*

Figure 5 illustrates the comparison between WebSphere Application Server, Version 4.0 and Version 3.5.3.

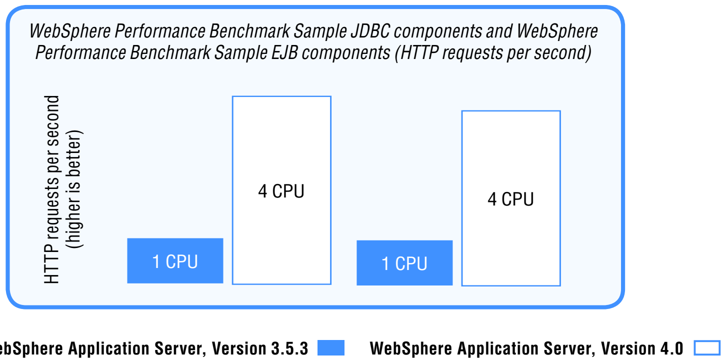


Figure 5. WebSphere Application Server, Version 4.0 compared to WebSphere Application Server, Version 3.5.3.

Twelve-way scaling

One of the performance focus areas for WebSphere Application Server, Version 4.0 is scalability on systems with a larger number of processors. Consequently, focus was placed on performance of WebSphere Performance Benchmark Sample EJB components running on 4- and 12-CPU systems.

Twelve-way scaling results

WebSphere Application Server, Version 4.0 has a higher throughput (up to 34 percent) at each processor configuration than WebSphere Application Server, Version 3.5.3.

Figure 6 compares WebSphere Application Server, Version 4.0 to Version 3.5.3 and shows its higher throughput.

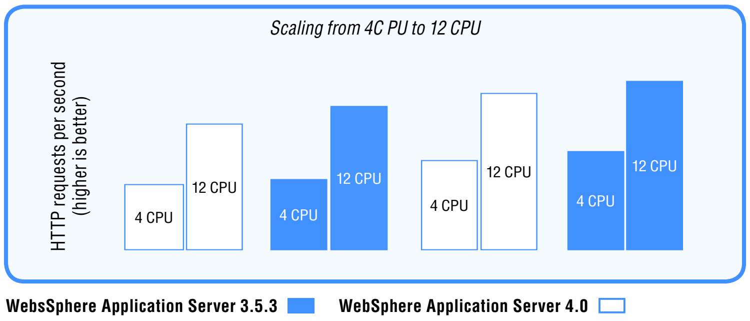


Figure 6. Higher throughput of WebSphere Application Server, Version 4.0.

Scaling on IBM AIX 24-way

WebSphere Application Server, Version 4.0 was also tested for its performance on 24-CPU systems using WebSphere Performance Benchmark Sample EJB components running on a 24-CPU IBM RS/6000® S85 system. At the time, the S85 was the highest performing AIX system available within IBM.

Scaling on IBM AIX 24-way results

Figure 7 shows the scalability of WebSphere Application Server, Version 4.0 on systems with 6 and 24 processors compared to Version 3.5.3. For WebSphere Performance Benchmark Sample JDBC components, Version 4.0 has a performance increase of 11 to 36 percent, depending on the operating system platform. For WebSphere Performance Benchmark Sample EJB components, Version 4.0 has a performance increase of 15 to 75 percent, depending on the operating system platform. In addition to performance, scalability has increased in Version 4.0 to 14.4/24 from the 9.2/24 ratio of Version 3.5.3. (A scaling ratio is the measurement that shows increased workload performance as the number of processors increases.)

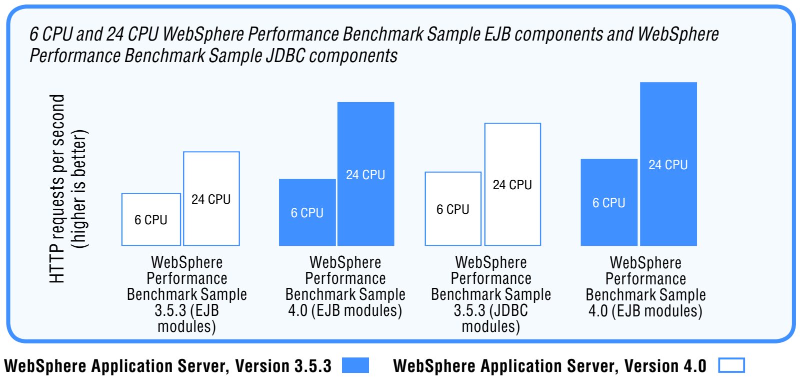


Figure 7. 6 CPU and 24 CPU WebSphere Performance Benchmark Sample EJB components and WebSphere Performance Benchmark Sample JDBC components.

WebSphere Edge Server builds on the performance features of WebSphere Application Server, Version 4.0

WebSphere Edge Server is the strategic platform from IBM for distributing application processing to network edges. The tight integration of the WebSphere Edge Server with the WebSphere family of products makes it an optimal vehicle to delay centralized applications by deploying selected components into the network.

The enhanced cache improves response time by offloading back-end servers and peering links as a forward proxy (intermediary), reverse proxy (surrogate) or transparent proxy, and it is programmable by using plug-ins. It surpasses industry standards in its innovative capabilities to cache and invalidate dynamic content generated by WebSphere Application Server, as well as caching fragments and variants. Test results involving simple JSP technology-based processing have shown that WebSphere Application Server performance can increase by 76.92 transactions per second (TPS) when used with WebSphere Edge Server dynacaching feature – an 82 percent increase over a WebSphere Application Server deployment without WebSphere Edge Server. And when coupled with SSL for encryption, WebSphere Application Server performance is increased by 22.38 TPS when used with the WebSphere Edge Server dynacaching feature for a 259 percent increase.

In addition, load balancing can further improve your Web site's availability, scalability and performance by transparently clustering edge, Web and application servers. It also provides site selection, workload management, session affinity and transparent fail-over. WebSphere Edge Server provides a unique custom advisor to load balance WebSphere Application Servers, which enables load balancing based upon selected application and platform criteria. You can use the content-based routing in WebSphere Edge Server in a variety of ways. For example, you can divert overloads to a service provider with excess capacity. WebSphere Edge Server provides many other capabilities as well, including edge-of-network security, content distribution, differentiated qualities of service that dynamically account for differing levels of transacted value.

Figure 8 shows the typical configuration of WebSphere Edge Server.

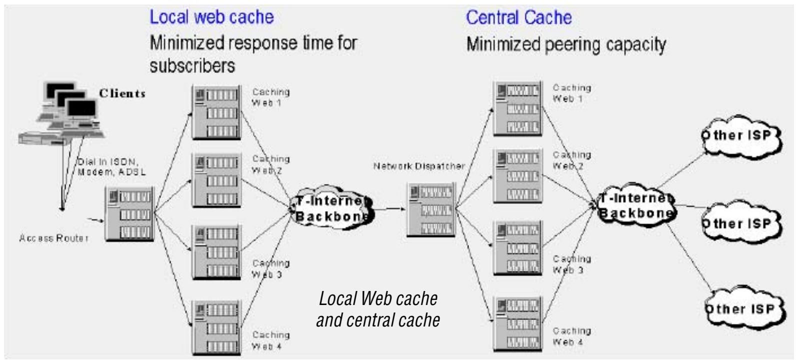


Figure 8. The typical configuration of WebSphere Edge Server.

Capacity planning

Capacity planning is crucial to getting your server to run at the optimal performance level. WebSphere Application Server, Version 4.0 has several tools to help you quickly and accurately estimate your projected workload and to come up with the most efficient configuration to meet your needs.

How to estimate server workload

There are three steps to this estimation technique:

1. Estimate the concurrent client (user) requirements
Ask the question: Given my Web application and the anticipated traffic, how many concurrent clients need to be serviced at a random point in time during peak traffic? That is, if you could freeze the server at a specific point in time and count the total number of client requests in the system, how many would there be?

2. Estimate total concurrent clients supported
Ask the question: Given a fixed server configuration, how many clients can the application support concurrently?

3. Estimate server configuration
Take the answers from steps 1 and 2 and decide which configuration is required.

In planning your Web site, you need to estimate the number of users per day of your Web site and the expected length of time each user will stay. For this example, your Web site will be supporting 100,000 users, and the estimated length of each user session is 5 minutes.

U = number of users per day to your application = 100,000
Y = minutes in a user session = 5

Next, plan for your site's anticipated daily peak usage. Although your site might have 100,000 users a day, these users are probably not spread evenly across that 24-hour period. Therefore, in any given hour, you might have to support more than 1/24 of the total daily traffic. If you are unsure how the usage will be spread throughout the day, you can estimate that ¼ of your daily usage might occur in any given hour.

Use the following formula to calculate an initial estimate of daily peak usage using:

U (users) = 100,000

Y (daily traffic*) = 5

60 = minutes per hour

(* 1/24 of total daily traffic or ¼ daily usage in any given hour)

To calculate one-fourth of your daily peak usage:

U/4 = peak users per hour (or 100,000 ÷ 4) = 25,000 users/sessions per hour (peak periods)

Each hour has 12 five-minute periods, therefore: 60 ÷ 5 = 12

To estimate the number of concurrent users/sessions during that 5-minute period

25,000 ÷ 12 = 2,000 concurrent users/sessions in a 5-minute span (approximately 2,000)

This calculation produces the number of concurrent clients you are going to have at your peak hour.

Another way to estimate the number of concurrent clients using the figures from the preceding example is to use the formula shown here, where *U = 100,000 users* and *Y = minutes in a user session*:

(U*Y)/240 = concurrent clients (estimated)

For this hypothetical Web site, the site's capacity plan should support approximately 2,000 concurrent sessions.

How to estimate total concurrent users (or clients) supported

To estimate the number of users, *or clients*, your server configuration can support, you will need to understand the performance characteristics of your application or a similar application. (The WebSphere Performance Benchmark Sample application, for example, supports approximately 100 concurrent active clients with a specific throughput and response time on a specific server.) The next step is to bridge from concurrent active clients to the number of clients supported.

What is the definition of *active* client? Active clients have both an active connection to the HTTP server as well as at least one thread of execution running in that server. At any point in time, many of the clients connected to a Web site are not active. These are called inactive clients.

If a server can support X concurrent active clients, how many total clients (active + inactive) is it capable of supporting? Based on experience, a good rule of thumb is to assume that for each active client, there are approximately 10 inactive clients.

Applying the preceding assumption, a server that supports 100 active clients will support approximately 1,000 concurrent clients (active + inactive). Therefore, the WebSphere Performance Benchmark Sample application configuration will support 1,000 concurrent clients.

How to estimate server configuration

In step 1 the concurrent client need for this Web site was estimated to be 2,000. In step 2, the WebSphere Performance Benchmark Sample application on the specific server configuration was estimated to support 1,000 total clients. If the site application is similar to WebSphere Performance Benchmark Sample, scaling alternatives need to be applied to support 2,000 clients with similar response time and throughput to the WebSphere Performance Benchmark Sample performance shown earlier. Since the WebSphere Performance Benchmark Sample server configuration supported 1,000 concurrent clients applying the scaling alternatives discussed earlier, it makes sense to plan for a cluster consisting of two such servers.

How to estimate memory

Memory consumption is another important component of capacity planning. To efficiently run the WebSphere Application Server, a minimum of 512MB of RAM per processor is recommended.

For more information about capacity planning, visit the IBM High Volume Web site capacity planning URL at:
www7b.boulder.**ibm.com**/wsdd/zones/hvws/library.html

IBM High Volume Web Site Simulator tool for WebSphere

The IBM High Volume Web Site team has developed the High Volume Web Site Simulator, an analytic queuing tool to estimate performance and capacity of Web site infrastructure. The goal of this team is to enable customers to design and deploy highly scalable, high-performance Web-based e-business solutions to meet their specific needs. Because configuration planning for large Web servers is becoming increasingly complex and the behavior characteristics of the workloads are often not fully understood, the team focuses on offering customer solutions that solve specific problems related to performance and capacity planning. The simulator is one such tool and is used to do capacity planning for WebSphere software-specific installations.

The tool is based on extensive measurements of typical customer workloads and best practice rules developed from customer benchmark experience. The simulator tool is used by IBM marketing and service professionals for the purpose of presales analysis of architecture and what if analysis of performance and capacity.

The High Volume Web Site Simulator for WebSphere offers the following features to simplify capacity planning:

- *Prebuilt e-business workload patterns. These built-in workloads are designed for primary business patterns, such as shopping, trading, banking and B2B transactions. The user can use these prebuilt workloads or modify them based on a customer's requirements.*
- *User-defined workload. If the prebuilt workloads do not meet the customer's workload, the user can substitute other measurements.*
- *Peak-usage algorithms. The simulator also provides special algorithms to handle bursts of Web traffic that can occur during peak usage periods, such as during a holiday season or a heavy stock market session. The user defines how heavy the bursts are in terms of peak/average ratio and then sets a specific performance objective for items such as user visit rate, resource utilization, response time, number of concurrent users and page view rate. The user can also define details of the n-tier topology and select an IBM or Sun system.*
- *The tool can also estimate various performance parameters such as throughput, response times, resource utilization and number of concurrent users. It will also graphically identify resource bottlenecks.*

The infrastructure supporting most high-volume Web sites typically has multiple components that include clients, the network and multiple layers of machines within the Web server. These multiple machine layers are frequently called tiers, with each tier handling a particular set of functions, such as serving content (Web presentation servers), providing integration business logic (Web application servers) or processing database transactions (database servers). While actual customer implementations can vary widely, the simulator uses the following generalized view of the infrastructure options shown in Figure 9.

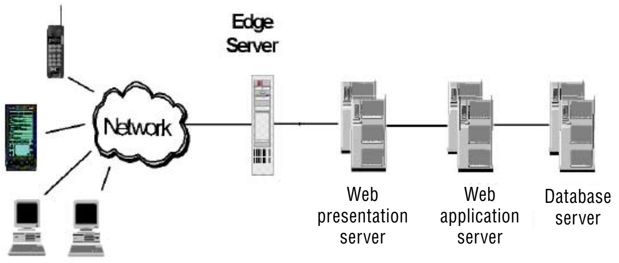


Figure 9. Topology of a three-tier Web server.

The High Volume Web Site team is available to help customers address the challenges of increasing growth of Web site traffic, as well as the growing complexity of e-business infrastructures. The team has worked with a diverse group of clients including Charles Schwab, Aetna, Toronto Dominion Bank, Fidelity, VISA, Bank of America, Barclays Bank, walmart.com, eBay and others. To obtain more information compiled by the High Volume Web Site team published as best practices white papers, visit: **ibm.com**/websphere/developer/zones/hvws

WebSphere Studio Application Developer enhances WebSphere Application Server, Version 4.0

IBM WebSphere Studio Application Developer is a premier, end-to-end J2EE application development tool optimized for the WebSphere software platform and represents the next generation of IBM offerings that make building e-business applications easier and faster. Based on WebSphere Studio Workbench technology, the application developer enables unprecedented tool integration, dramatic ease of use and industry-leading support for open technologies.

The application developer is designed for professional developers of Java and J2EE applications, requiring integrated Web, XML, Web services and JSP technology-based support. It includes a highly integrated WebSphere Application Server test environment and deployment automation tools, plus advanced tools for code generation and performance tuning.

WebSphere Studio Application Developer is next-generation technology for VisualAge for Java, Enterprise Edition. It combines many popular features of VisualAge for Java, Enterprise Edition and WebSphere Studio Advanced Edition integrated on top of WebSphere Studio Workbench.

This application developer is focused on optimizing J2EE application development. Java developers can create, test, and deploy JavaBeans components, EJB components, JSP files and servlets. It is based on open standards to meet requirements for the next generation of Web application development. WebSphere Studio Application Developer also allows independent software vendors (ISVs) to easily integrate their solutions into the application developer.

Key features include support for:

- *Building J2EE applications with HTML pages, servlets, JSP files and EJB components*
- *Creating Web-service applications with open standards*
- *Generating XML documents from DTDs, schemas*
- *Enabling a collaborative team environment*
- *Optimizing application performance*
- *Enabling end-to-end local and remote testing*
- *Increasing productivity and creating high-quality applications using wizards, code generators and best practices*

Compatible operating systems:

- *Windows 98*
- *Windows Me*
- *Windows NT® Workstation/Server Version 4.0, SP6a (or higher)*
- *Windows 2000 Professional/Server/Advanced*

The WebSphere Studio Application Developer includes the following functions:

- *An advanced development environment for J2EE application development that meets J2SE and J2EE specifications, EJB modules development and deployment and supports visual layout of dynamic pages, full HTML, JavaScript and DHTML*
- *A powerful Java development environment that includes support for IBM Developer Kit for the Java Platform, Version 1.3, a configurable runtime, incremental compilation, scrapbook, dynamic debugging and Java text editor*
- *Tools to create, build, deploy, test, discover and publish Web-service-enabled applications, including support for Universal Description Discovery and Integration (UDDI), Simple Object Access Protocol (SOAP) and Web Services Description Language (WSDL)*
- *A relational schema center (RSC) focused on relational database design and database administration tasks, such as import and mapping schemas and other advanced functions*
- *Application performance profiling and trace code to identify, isolate and fix code*
- *An integrated roles-based team environment that allows for flexible source-control management, pluggable repositories, as well as adapters for Concurrent Versioning System (CVS) and for Rational ClearCase LT (CCLT)*
- *A comprehensive visual XML development environment, including components for building DTDs, XML schemas, XML files, XSL style sheets, SOAP and WSDL*
- *Built-in WebSphere test environment*

Summary

WebSphere Application Server Version 4.0 provides a premier environment for robust and high-performance Java technology-based Web applications including servlets, EJB components and JSP files . The new environment of the 4.0 release improves on the 3.5.3 release and is competitive to traditional Internet-standard CGI-based applications and alternative Enterprise Java application servers.

The performance of the WebSphere Application Server is best examined by end-to-end customer scenarios. The WebSphere Performance Benchmark Sample and the ECperf Benchmark Kit provide mechanisms for characterizing performance in both an end-to-end customer scenario and a series of resulting component benchmarks. WebSphere Application Server interoperates with other components of the WebSphere and IBM middleware family (including IBM CICS® and DB2) to provide a competitive solution for building high-performance e-business applications. In 2002, IBM will continue to improve the performance characteristics of WebSphere Application Server by drawing on the company's vast experience in high-performance e-business systems.

For more information

IBM is the world's largest information technology company, with 80 years of leadership in helping businesses innovate. IBM software offers the widest range of infrastructure software for all types of computing platforms, allowing customers to take full advantage of the new era of e-business. To get more information about IBM software, visit **ibm.com**/software.

Find out more about WebSphere performance by accessing white papers, best practices documents, Redbooks™, various tool guides and downloads at the IBM WebSphere Application Server performance Web site at:
www-3.**ibm.com**/software/webservers/appserv/performance.html

For more info about WebSphere ECperf benchmark results, visit:
http://ecperf.theserverside.com/ecperf/index.jsp?page=results/top_ten_price_performance

For more information about IBM WebSphere Commerce Suite, Pro Edition, visit:
http://webspherepro.com wparchiveindex.cfm?fuseaction=viewarticle&ContentID=439

WebSphere Application Server, Version 4.0 customer case studies and press announcements

To access WebSphere Application Server, Version 4.0 customer case studies, visit:
www-3.**ibm.com**/software/webservers/appserv/cust_quotes.html

To read WebSphere Application Server, Version 4.0 press announcements, visit:
www-3.**ibm.com**/software/webservers/appserv/news/kana.html

IBM WebSphere Application Server wins PC Magazine Editors' Choice award

IBM WebSphere Application Server has been recognized for its excellent performance by a variety of independent agencies. In the spring of 2001, *PC Magazine* hosted benchmarks to test different application servers. Dynamic caching technology played a key role in WebSphere winning the magazine's Editors' Choice award for fastest and most scalable application server. Read the entire *PC Magazine* article by visiting:
http://www.zdnet.com/products/stories/reviews/0,4161,2713476,00.html

e business software

G325-5543-00