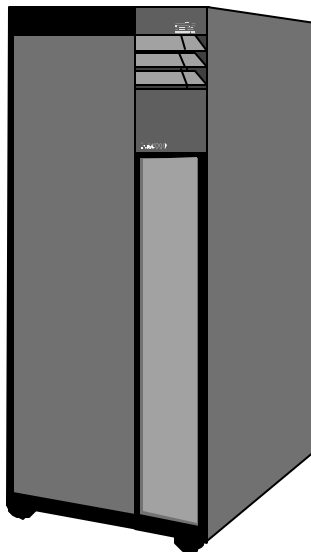


**IBM Corporation**

**A Case Study in WebSphere Application Server Performance on the  
IBM @server pSeries 660 Model 6M1**

**Tony Garcia**

**February 10, 2002**



## Table of Contents

<b>Overview</b> .....	3
<b>Executive Summary</b> .....	3
<b>Trade 2 Benchmark</b> .....	3
<b>System Configuration</b> .....	3
<b>Hardware Configuration</b> .....	4
<b>Software Levels</b> .....	5
<b>AIX</b> .....	5
<b>DB2</b> .....	6
<b>WebSphere</b> .....	6
<b>IBM HTTP Server</b> .....	6
<b>Trade 2</b> .....	6
<b>Test Procedure</b> .....	6
<b>Trade 2 Results</b> .....	7
<b>Summary</b> .....	7
<b>CPU Utilization</b> .....	8
<b>Garbage Collection</b> .....	9
<b>Hard Disk Utilization</b> .....	9
<b>Network Usage</b> .....	9
<b>Conclusions</b> .....	9
<b>References</b> .....	11

## Executive Summary

This report details the results of an IBM study of the WebSphere® Application Server version 4.0 running under AIX® 5L™ version 5.1 on an IBM @server pSeries™ 660 Model 6M1. The internal benchmark Trade 2 was used to measure the performance of the p660-6M1.

This report discusses the environment and the final results achieved in requests per second for a workload running in a two-tier environment. A run-rule of 100 users making 500,000 requests of the WebSphere server was used.

The system tested was an IBM @server pSeries 660 Model 6M1 running at 750 MHz with 8 RS64 III processors, 16 GB of memory (RAM), a total of 8 SSA disk drives on a rack enclosure divided into two loops, a built-in 10/100 Ethernet adapter, a Gigabit Ethernet card and a single internal SCSI drive to hold the AIX operating system. The machine used to drive the workload is a single Netfinity 6000R with 4 700 MHz Pentium® III processors with 2 GB of memory (RAM), a single SCSI drive to hold the Red Hat Linux® 6.2 operating system and a Gigabit Ethernet card connect to the pSeries 660 Model 6M1 via a point-to-point connection.

Testing showed excellent scaling up to an 8-way configuration. The results are as follows:

# CPUs	EJB req / sec	Scaling	JDBC req / sec	Scaling
1-way p660	258.8	x	342.23	x
4-way p660	934.58	3.61	1,262.63	3.69
8-way p660	1,666.66	6.44	2,242.13	6.55

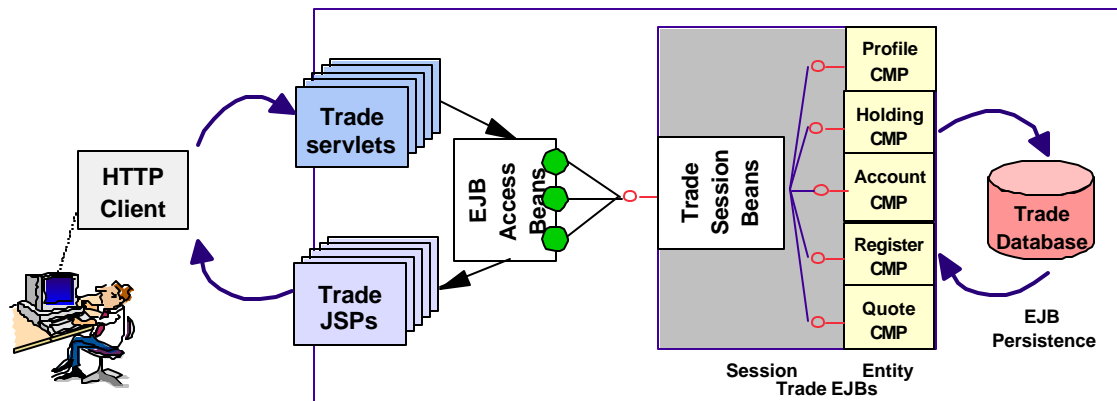
## Overview

The pSeries 660 Model 6M1 is a powerful up to 8-way symmetrical multiprocessor server. It is highly adaptable and as such, is appropriate for a variety of environments and applications. With Web-based services growing exponentially and the dependence on Web-based transactions increasing, it is important to be able to trust your server. The pSeries 660 Model 6M1 running AIX 5.1.0 maintenance level 01 and the WebSphere Application Server Advanced Edition Version 4.0.1, the WebSphere Application Server Advanced Edition V4.01, and DB2® Enterprise Edition V7.2 is a great combination for customers. Based on research conducted using the Trade 2 benchmark, the pSeries 660 Model 6M1 8-way demonstrates excellent throughput and scaling. This makes the p660 Model 6M1 a viable solution in today's e-business world. This paper is an update to the April 17, 2001 paper "A Case Study in WebSphere Application Server Performance on the IBM RS/6000® Model M80" and offers a comparison study to the work detailed in that paper.

## Trade 2 Benchmark

The Trade 2 benchmark, also called the WebSphere Performance Benchmark Sample, developed by IBM and is publicly available, measures the performance of servers running the IBM WebSphere Application Server software. Trade 2 was built to emulate an online brokerage firm. This workload exercises the entire solution stack that consists of the WebSphere Application Server, the Java Virtual Machine™ (JVM) and the Just-In-Time (JIT) compiler, the HTTP server, the DB2 Database Server and the DB2 client, the AIX operating system, and the system hardware.

The Trade 2 application is a collection of Java™ classes, Java Servlets, Java Server Pages and Enterprise Java Beans (EJBs) that service requests made by registered users. This application runs as a single java process which is managed by the WebSphere Application Server.



Trade 2 performance is measured by how many requests the server can handle per second (throughput). This is accomplished by using a HTTP request generator to simulate multiple users making hundreds of requests to the server.

For the purposes of this research, the internally developed “akstress” tool was used to generate the requests using a Trade 2 supplied script for the user simulation. The Trade 2 servlet on the WebSphere server has a function whereby it randomly generates an access to the server. Akstress generates hits to this servlet via Web requests and then generates a report based on the results it received from the server. Results are displayed in Requests per Second. Akstress was configured to run with 100 threads per run as defined by the run rules.

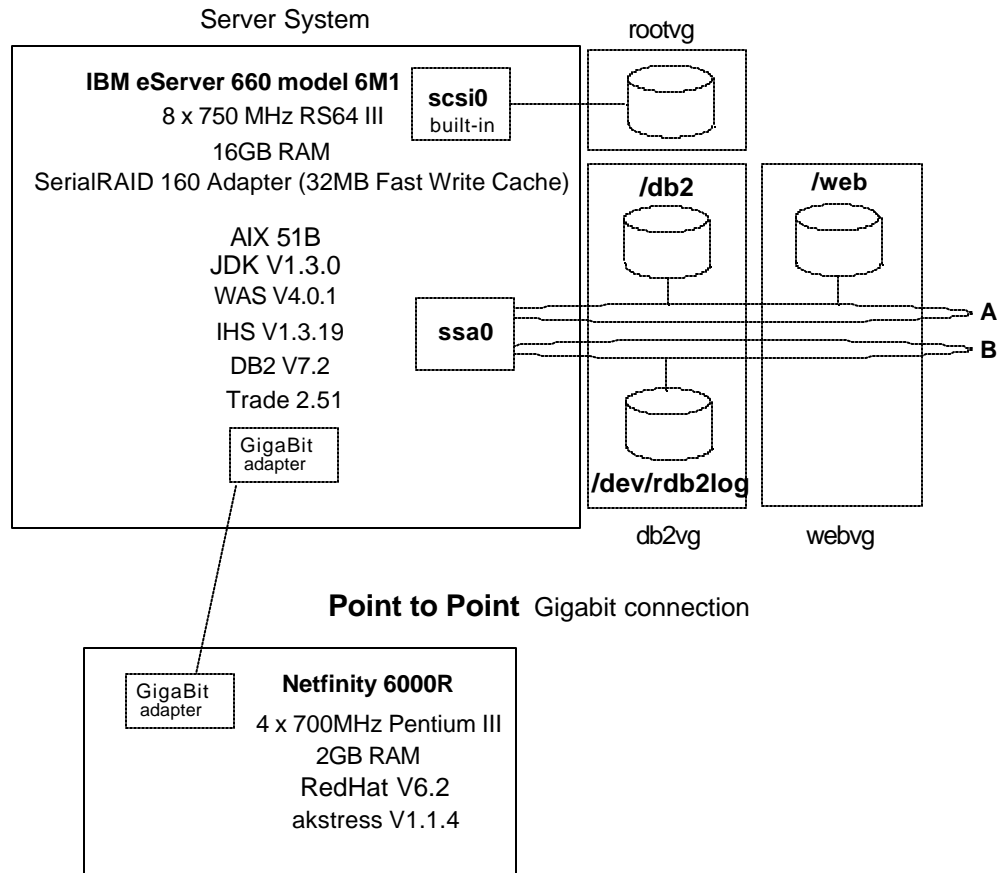
## System Configuration

The test system was setup as a two-tier environment. Tier 2 consists of the WebSphere Application Server, IBM HTTP web server and the DB2 database server on the same machine. Tier 1 consists of the machines used to drive the system. The driver used was a IBM Netfinity 6000R 4-way machine running the Linux Red Hat 6.2 operating system. A single Linux driver running the akstress software configured to simulate 100 users (100 threads) was used for testing both EJB, Enterprise Java Beans, and the JDBC, Java DataBase Connection, methods of accessing the DB2 database.

For connectivity, a Gigabit network adapter was installed on both the Netfinity 6000R and the p660 and were connected using the Point-to-Point method. A 100 Mbps Ethernet card was also installed in the p660, but was only used for the software installation of the WebSphere Server, DB2 and other performance tools.

### Hardware Configuration

The p660 model 6M1 was configured with 8 x 750 MHz RS64 III processors, 16 GB of memory (RAM), a total of 8 SSA disk drives divided into two loops, a built-in 10/100 Ethernet adapter, a Gigabit Ethernet card and a single internal SCSI drive to hold the AIX operating system. The system was configured to ensure that the limiting factor (bottleneck) is the CPU. Where possible, each software package was installed on separate physical disks with a separate volume group and file system to prevent IO from becoming a bottleneck.



To ensure stability of the SSA subsystem, it is recommended that the AIX device drivers and the microcode for the SSA components are updated to the latest level. The latest levels of SSA software can be downloaded from <http://www.rs6000.ibm.com/support/micro/ssa.html>.

After installing the software on the system, a number of commands must be run to force the microcode updates to the adapter(s), disks and the controller in the disk enclosure, while there is no I/O to the SSA subsystem (database manager is stopped and the file systems are unmounted).

#### Software Levels

```
AIX V5.1.0 ML 01 (5.1.0 + IY21957
ptf=u480124 )
IBM DB2 Universal Database V7.2
IBM HTTP Server V1.3.19 for AIX
JDK V1.3.0 (ships with WAS V4.0.1)
WAS Advanced Server V4.01
Trade 2 (V2.51)
```

To check the levels of the individual software components, run the following commands:

AIX	oslevel lslpp -l bos.rte lslpp -l bos.mp	5.1.0 5.1.0.10 5.1.0.10
DB2	lslpp -l db2*	7.1.0.43
IHS	/usr/HTTPServer/bin/httpd -v	1.3.19
JDK	export WAS_HOME=/web/WebSphere/AppServer export JAVA_HOME=\$WAS_HOME/java export PATH=\$JAVA_HOME/jre/sh:\$JAVA_HOME/sh:\$PATH which java java -version	ca130-20010615a
WAS	grep -i number \$WAS_HOME/properties/com/ibm/websphere/product.xml	4.0.1 (a0131.07 build)
Trade 2	<a href="http://&lt;hostname&gt;/trade/index.html">http://&lt;hostname&gt;/trade/index.html</a>	Trade 2.51

## AIX

AIX version 5.1.0 (AIX 5.1.0 ML 01) was installed on the 6M1 internal SCSI drive. Only those components necessary to run AIX, WebSphere and DB2 are installed on the machine. In addition, all but the following are removed from the /etc/inittab file.

```
init:2:initdefault:
brc::sysinit:/sbin/rc.boot 3 >/dev/console 2>&1 # Phase 3 of system boot
rc:23456789:wait:/etc/rc 2>&1 | alog -tboot > /dev/console # Multi-User
checks
fbcheck:23456789:wait:/usr/sbin/fbcheck 2>&1 | alog -tboot > /dev/console
# run /etc/firstboot
rctcpip:23456789:wait:/etc/rc.tcpip > /dev/console 2>&1 # Start TCP/IP
daemons
12:2:wait:/etc/rc.d/rc 2
13:3:wait:/etc/rc.d/rc 3
14:4:wait:/etc/rc.d/rc 4
15:5:wait:/etc/rc.d/rc 5
16:6:wait:/etc/rc.d/rc 6
17:7:wait:/etc/rc.d/rc 7
18:8:wait:/etc/rc.d/rc 8
19:9:wait:/etc/rc.d/rc 9
ctrmc:2:once:/usr/bin/startsrc -s ctrmc > /dev/console 2>&1
lft:2:respawn:/usr/sbin/getty /dev/lft0
```

This minimizes the number of processes running in the background to only those that are absolutely necessary. Certain AIX variables are also set in order to improve performance of the system.

```
export AIXTHREAD_COND_DEBUG=OFF
export AIXTHREAD_MUTEX_DEBUG=OFF
export AIXTHREAD_RWLOCK_DEBUG=OFF
export AIXTHREAD_SCOPE=S
unset MALLOCMULTIHEAP
unset SPINLOOPTIME
```

These are environmental variables that can be set in the .profile and should be set before any software is run on the system. MALLOCMULTIHEAP and SPINLOOPTIME are unset by default in AIX. Unsetting them here ensures that they are not being used. Any changes made to these variables require a stop and restart of the applications in order for the changes to take effect.

## DB2

DB2 was configured to use three SSA drives spread across two SSA loops. All drives are part of the same volume group (db2vg). The log for the db2vg volume group and the file system, /db2 for storing the databases are located on the first drive. The other two drives are used as a raw logical volume for the DB2 logs and are on a different SSA loop from the first drive. The data on these two drives is striped across both drives for better performance. For a real customer environment, the databases and logs would be spread over a lot more drives.

The two databases (wasdb and tradedb) were created and accessed using the aliases was and trade respectively.

```
db2 catalog tcpip node node1 remote <hostname> server <db2_service_port>
db2 create db wasdb
db2 catalog db wasdb as was at node node1
db2 create db tradedb
db2 catalog db tradedb as trade at node node1
```

The loopback configuration is used here to provide a similar test environment to compare with the earlier M80 test. You can do without the loopback configuration with DB2 V7.2 or higher, on AIX 4.3.3 or higher, by doing the following before starting up DB2:

```
export EXTSHM=ON
db2set DB2ENVLIST=EXTSHM
```

There are two databases used by this benchmark. The WebSphere configuration information is stored in the "was" database which is of minimal interest or worry since it is only accessed during initialization. The application data is stored in the "trade" database, containing information about the user accounts of the simulated brokerage house for the Trade 2 application. This database has an initial size of 12 megabytes, but grows and shrinks as user's stocks are bought and sold during the course of the Trade 2 test run.

DB2 has a number of configuration settings for each database, as well as the database manager instance. The following settings were applied to the trade database.

```
applheapsz 256
buffpage 40000
(requires db2 alter bufferpool ibmdefaultbp size -1 to take effect)
maxappls 256
avg_appls 256
maxlocks 75
logbufsz 512
```

The important step is to perform runstats against the trade database after it has been populated to leverage the DB2 optimizer.

### WebSphere

The WebSphere Application Server Version 4.0.1 is installed on a separate physical disk. In order to achieve the best results the Prepared statement cache for the Trade 2 database must be set. For the purposes of our test we have set the statement cache to 2000. The number of WebSphere threads is set to 50 and the number of DB2 threads is set to 100. The Pass-by-Reference option was also used.

### IBM HTTP Server

The IBM HTTP Server (IHS) Version 1.3.19 for AIX was used as the web server. The installation procedure for the WebSphere Application Server provides an option to select the plug-in module for this web server. The only changes made to the httpd.conf file for these tests are:

```
ServerName heracles
MinSpareServers 5
MaxSpareServers 107
StartServers 107
```

Setting the number of server processes is not recommended for customer environments. The only reason for setting them here is to maintain a constant number of httpd processes between runs as we know that we will be hitting the server with 100 connections (one httpd process for each connection).

By default, IHS will be installed in the `/usr/HTTPServer` directory so all the logs for the web server will be located in the root volume group (rootvg). The only file that should be monitored for growth is `/usr/HTTPServer/logs/access_log`. For the purposes of this test, both the `access_log` and the `error_log` function of the HTTP server were disabled. This reduces the contention on the `inode_lock` for these files. However, due to security concerns this is not recommended for customer environments.

## Trade 2

For these tests, Trade 2 is run with Java min and max heap size of 1024M (1 GB). This reduces the frequency of garbage collection but incurs the cost of longer garbage collection sessions. The stack size is also set to 1024K allowing for a greater number of instructions to be placed on the stack for execution. The stack size settings can be set by selecting the server, and then choosing the JVM Settings tab. WebSphere provides two box options for this. The stack size can be added by pressing the Advanced options button in the JVM Settings screen and then the following can be added to the options field:

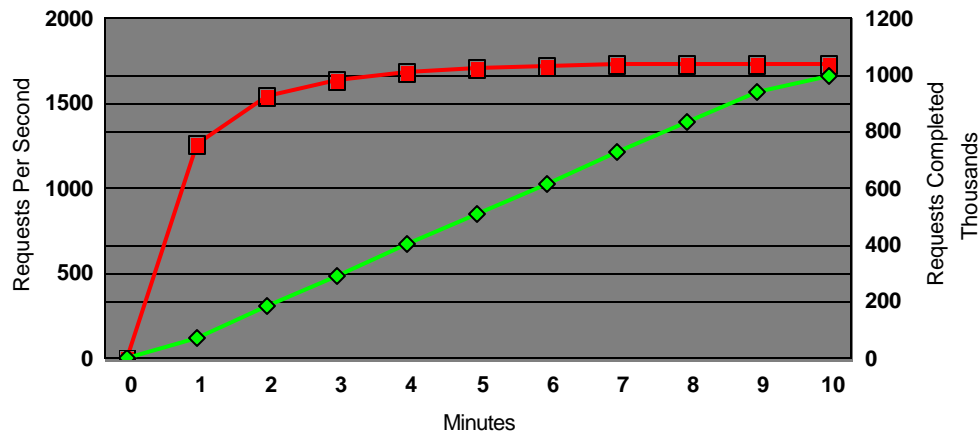
```
-ss1024k
```

The statistics for garbage collection can be collected by selecting the "Enable garbage collection verbose mode" radio button in the Advanced Option page in the JVM Settings command line. The output is added to the standard error file for the Trade 2 application. You can view the output during the Trade 2 runs using:

```
tail -f $WAS_HOME/logs/Default_server_stderr.txt
```

## Test Procedure

The Trade 2 benchmark functions in several different modes of operations. For the purposes of this document only two modes were used, EJB and JDBC. The difference is really how the database is being accessed. For the EJB mode, Trade 2 uses the VisualAge for Java EJB access beans to access the trade database. For the JDBC mode, the database access uses direct JDBC code, with no EJBs. Well written



JDBC code will generally outperform EJB database access code.

Both modes accept a request from the user and then handle that request. Several actions must then occur. The request is made by an end-user, using a browser he makes his selections and submits the request. The request is handled by the HTTP web server which hands it over to the WebSphere Application Server. The WebSphere Application Server then performs the action, usually accessing and updating the Database. Then returns its response to the HTTP web server and finally returns to the end-user. Over time the database grows and shrinks in accordance with the requests made. For these tests, the database was restored to its initial state before each run.

The following graph illustrates the ramp-up in throughput (Requests per Second) over time for a 1 Million request run (EJB Mode) against the 8-way p660 model 6M1. The initial delay in throughput is due to the Java classes being loaded. The throughput begins to climb and then starts to level off as it approaches the maximum throughput of the system. The throughput remains stable for the remainder of the test as the Just-In-Time (JIT) compiler has completed compilation of all the frequently used methods. The data shows that Trade 2 does not reach a steady state until it completes 180,000 requests. Because of this the akstress driver system is configured to simulate 100 users making a total of 500,000 requests against the server.

## Trade 2 Results

### Summary

Here is the summary of the Trade 2 results for the pSeries 660 Model 6M1 using:

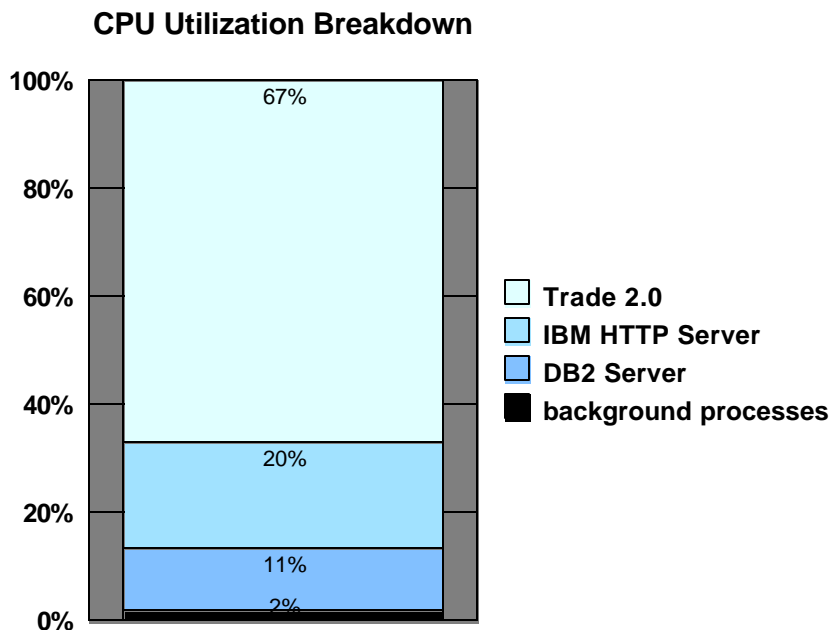
- 500K Total requests
- 1 GB min/max Java heapsize
- 1 MB Stacksize
- 50 WebSphere Container Threads
- 20 ORB Threads
- 100 Database connection pool size
- Pass by Reference
- 107 HTTP Threads
- 2000 Prepared Statement Cache

# CPUs	req/sec EJB-mode	Scaling	req/sec JDBC-mode	Scaling
1-way p660	258.8	x	342.23	x
4-way p660	934.58	3.61	1,262.63	3.69
8-way p660	1,666.66	6.44	2,242.13	6.55

The Trade 2 results are the average number of HTTP requests/sec as reported by the akstress HTTP request driver running against the Trade servlet configured in either EJB or JDBC mode for the entire run (500K total requests using 100 threads). For the Trade 2 workload, each request is a page, so requests/sec is equal to pages/sec.

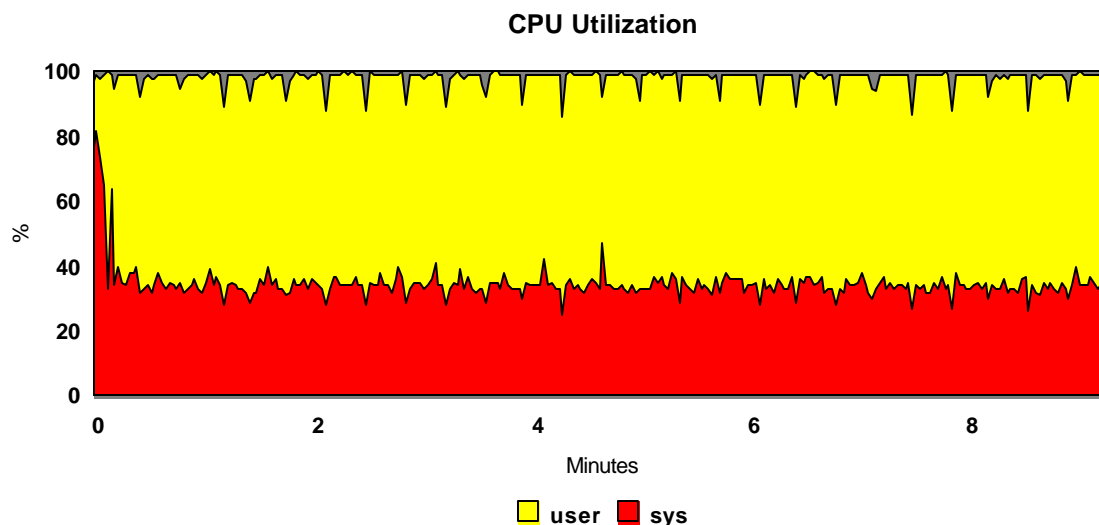
### CPU Utilization

This graph shows a snapshot of the system utilization in between garbage collection cycles. The most active process during a run belongs to the Trade 2 application process. There are two additional Java processes for the nanny process and the admin server but they do not register any CPU after the initial launch of the application. The HTTP server processes (httpd) consumes 20% of the system. The DB2 server uses 11% of the system.



## Garbage Collection

For any Java programming environment, garbage collection is a sensitive issue that has received a lot of interest and attention. The following graph shows the effect on the CPU utilization due to garbage collection. The graph does not accurately show the CPU utilization during the garbage collection cycle as the sampling rate for CPU is once per second and the garbage collection takes less than a second.



The garbage collection is performed by a multithreaded garbage collection algorithm. This allows for more efficient garbage collection. The first garbage collection cycle also frees any unused Java classes. For the EJB test with 8 CPUs configured and a 1GB Java heap, each garbage collection cycle was 443-620 msec in duration and occurred every 17-18 seconds. By reducing the heap size, the frequency would increase but the garbage collection cycle would be shorter.

The following table shows a summary of the garbage collection statistics for the two modes - EJB and JDBC, using a 1 GB Java heap. The duration of garbage collection is in the same range, regardless of the number of available CPUs. The intervals between garbage collection cycles are shown for the number of CPUs configured.

# CPUs	EJB	JDBC
Duration	443-620 msecs	607-654 msecs
1-way	152 secs	269 secs
4-way	40secs	72 secs
8-way	21 secs	39 secs

For any application, garbage collection affects the response time for the end-user. If the transaction is in flight during garbage collection, then the delay due to garbage collection is added to the response time. Most application developers use metrics for the average response time and the maximum response time to the end users.

## Hard Disk Utilization

As stated before, the database is spread across three different disks. Usage for the disks containing the raw logical volume is between .4% to 3.1% busy. The average kilobytes per second usage for the actual database disk was between .5% to 2.3%.

## Network Utilization

Network usage is characterized by a transfer rate of 121 Mbits per second with 10 Mbits per second for the receive rate. Combining these two rates yields a rate slightly higher than the theoretical limit possible

for 100 Mbps Ethernet card, but still significantly lower than the theoretical limit for a gigabit Ethernet card. The network throughput averages about 14,000 packets per second, with the average packet size of 1015 bytes for transfer packets and an average of 101 bytes for receive packets.

## Conclusions

As demonstrated in this report, the IBM @server pSeries p660 model 6M1 makes an excellent choice for customers seeking a mid-range server for running the WebSphere Application Server, Advanced Edition V4.0.1 and DB2 Enterprise Edition V7.2. The p660 model 6M1 is capable of running several software servers at once and still scale well. We have demonstrated a 6.44x scaling factor for EJB processes and 6.55x factor for JDBC process running the Trade 2 benchmark. The p660 model 6M1 shows excellent scaling at an 8-way configuration. It performs at a high throughput level and is stable for large request runs. In addition, in comparison to the previous paper "A Case Study in WebSphere Application Server Performance on the RS/6000 M80" we have demonstrated that in migrating current systems in this environment has yielded a significant improvement. A 1-way system shows a 77% improvement, a 4-way system shows 72% improvement and the 8-way system shows a 64% improvement over the previous paper's results.

## References

WebSphere V4 Performance Tuning Guide - SG24-6176-00

DB2 UDB V7.1 Performance Tuning Guide - SG24-6012-00

© International Business Machines Corporation 2002

IBM Corporation  
Marketing Communications  
Server Group  
Route 100  
Somers, NY 10589

Produced in the United States of America  
02-02 All Rights Reserved

More details on IBM UNIX® systems, software and solutions may be found at  
[ibm.com/servers/eserver/pseries](http://ibm.com/servers/eserver/pseries)

The [e(server) server] brand consists of the established IBM e-business logo followed by the descriptive term "server".

IBM, the IBM logo, AIX, AIX 5L, DB2, DB2 Universal Database, pSeries, RS/6000, Netfinity and WebSphere are registered trademarks or trademarks of the International Business Machines Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company, Limited.

Intel and Pentium are registered trademarks and Pentium III Xeon are trademarks of Intel Corporation in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

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

IBM may not offer the products, programs, services or features discussed herein in other countries, and the information may be subject to change without notice.

General availability may vary by geography.

IBM hardware products are manufactured from new parts, or new and used parts. Regardless, our warranty terms apply.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Any performance data contained in this document was determined in a controlled environment. Results obtained in other operating environments may vary significantly.