IBM Power Systems with IBM i
Performance and Tuning Tips for Oracle’s
JD Edwards EnterpriseOne 9.0 and 9.1

John Brock
IBM i ERP Development Support
and
Clark Scholten
IBM Advanced Technical Skills – Oracle Applications

July 2013
# Table of contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change history</td>
<td>2</td>
</tr>
<tr>
<td>Executive Summary</td>
<td>3</td>
</tr>
<tr>
<td>Performance Management for JD Edwards EnterpriseOne</td>
<td>3</td>
</tr>
<tr>
<td>General Power Systems server with i Performance</td>
<td>4</td>
</tr>
<tr>
<td>JD Edwards EnterpriseOne Application Tuning</td>
<td>5</td>
</tr>
<tr>
<td>Measure and Analyze</td>
<td>5</td>
</tr>
<tr>
<td>Adjust</td>
<td>6</td>
</tr>
<tr>
<td>1) Multi-threaded kernels in JDE.INI - thread settings and call object kernels</td>
<td>9</td>
</tr>
<tr>
<td>2) Single-threaded kernels in JDE.INI - number of call object kernels</td>
<td>12</td>
</tr>
<tr>
<td>3) Maximum users for the JAS server in JAS.INI</td>
<td>12</td>
</tr>
<tr>
<td>4) JDENET connections for the JAS server in JAS.INI</td>
<td>13</td>
</tr>
<tr>
<td>4) Optimize the use of SQL Packages</td>
<td>14</td>
</tr>
<tr>
<td>5) Controlling the Priority and Pool of QSQRVR jobs</td>
<td>17</td>
</tr>
<tr>
<td>6) Adding a Low Priority Job Queue</td>
<td>21</td>
</tr>
<tr>
<td>Create new subsystem</td>
<td>21</td>
</tr>
<tr>
<td>Create new class</td>
<td>22</td>
</tr>
<tr>
<td>Create job queue</td>
<td>22</td>
</tr>
<tr>
<td>Modify subsystem</td>
<td>22</td>
</tr>
<tr>
<td>Add job queue to JD Edwards EnterpriseOne</td>
<td>23</td>
</tr>
<tr>
<td>Sending UBEs to new job queue</td>
<td>24</td>
</tr>
<tr>
<td>Validating the setup</td>
<td>26</td>
</tr>
<tr>
<td>Appendix A – Advanced Performance Tuning</td>
<td>27</td>
</tr>
<tr>
<td>IBM iDoctor for IBM i</td>
<td>27</td>
</tr>
<tr>
<td>Oracle’s Performance Workbench</td>
<td>27</td>
</tr>
<tr>
<td>Appendix B – Additional Information</td>
<td>28</td>
</tr>
<tr>
<td>Contact information</td>
<td>28</td>
</tr>
<tr>
<td>References</td>
<td>28</td>
</tr>
<tr>
<td>General System Tuning</td>
<td>28</td>
</tr>
<tr>
<td>Advanced Performance Tools</td>
<td>28</td>
</tr>
<tr>
<td>JD Edwards EnterpriseOne</td>
<td>28</td>
</tr>
<tr>
<td>Oracle Redpapers (userid and password required)</td>
<td>29</td>
</tr>
<tr>
<td>Additional Resources</td>
<td>29</td>
</tr>
<tr>
<td>Trademarks and special notices</td>
<td>30</td>
</tr>
</tbody>
</table>
Table of figures

<table>
<thead>
<tr>
<th>Figure Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>General kernel definitions</td>
<td>6</td>
</tr>
<tr>
<td>Figure 2</td>
<td>Call Object (BSFNs) Stats</td>
<td>7</td>
</tr>
<tr>
<td>Figure 3</td>
<td>Call object kernel settings</td>
<td>8</td>
</tr>
<tr>
<td>Figure 4</td>
<td>Security kernel settings</td>
<td>8</td>
</tr>
<tr>
<td>Figure 5</td>
<td>Network kernel settings</td>
<td>9</td>
</tr>
<tr>
<td>Figure 6</td>
<td>Multithreaded call object kernel settings</td>
<td>10</td>
</tr>
<tr>
<td>Figure 7</td>
<td>Thread detail screen for call object kernels</td>
<td>11</td>
</tr>
<tr>
<td>Figure 8</td>
<td>Using iSeries Navigator for thread count in call object kernels</td>
<td>11</td>
</tr>
<tr>
<td>Figure 9</td>
<td>WRKACTJOB and F11 twice for number of threads in call object kernels</td>
<td>12</td>
</tr>
<tr>
<td>Figure 10</td>
<td>Max Users settings</td>
<td>13</td>
</tr>
<tr>
<td>Figure 11</td>
<td>Pool Size settings</td>
<td>13</td>
</tr>
<tr>
<td>Figure 12</td>
<td>Monitor JDENET Connection Pool settings</td>
<td>14</td>
</tr>
<tr>
<td>Figure 13</td>
<td>Database Configuration Panel</td>
<td>16</td>
</tr>
<tr>
<td>Figure 14</td>
<td>JDBJ Connection Pools settings</td>
<td>18</td>
</tr>
<tr>
<td>Figure 15</td>
<td>JDBJ Pooled Database Connections details</td>
<td>19</td>
</tr>
<tr>
<td>Figure 16</td>
<td>Maximum Jobs in a Subsystem</td>
<td>20</td>
</tr>
<tr>
<td>Figure 17</td>
<td>Pool Data from Performance Tools System Report</td>
<td>21</td>
</tr>
<tr>
<td>Figure 18</td>
<td>Create subsystem description command</td>
<td>22</td>
</tr>
<tr>
<td>Figure 19</td>
<td>Create class command</td>
<td>22</td>
</tr>
<tr>
<td>Figure 20</td>
<td>Add routing entry command</td>
<td>23</td>
</tr>
<tr>
<td>Figure 21</td>
<td>Job queue revisions</td>
<td>23</td>
</tr>
<tr>
<td>Figure 22</td>
<td>OMW version detail</td>
<td>24</td>
</tr>
<tr>
<td>Figure 23</td>
<td>Job scheduler batch overrides</td>
<td>24</td>
</tr>
<tr>
<td>Figure 24</td>
<td>RUNUBE command</td>
<td>25</td>
</tr>
<tr>
<td>Figure 25</td>
<td>Version prompting screen</td>
<td>25</td>
</tr>
<tr>
<td>Figure 26</td>
<td>Advanced version prompting screen</td>
<td>26</td>
</tr>
<tr>
<td>Figure 27</td>
<td>Work with active jobs screen</td>
<td>26</td>
</tr>
</tbody>
</table>
## Change history

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Editor</th>
<th>Editing description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>June 22, 2009</td>
<td>Gerrie Fisk</td>
<td>Original version</td>
</tr>
<tr>
<td>2.0</td>
<td>November 16, 2011</td>
<td>Diane Webster</td>
<td>Added sections for SQL Packages and QSQRV jobs</td>
</tr>
<tr>
<td>3.0</td>
<td>March 19, 2012</td>
<td>Diane Webster</td>
<td>Added sections on low priority job queues</td>
</tr>
<tr>
<td>4.0</td>
<td>July 9, 2013</td>
<td>Clark Scholten</td>
<td>Updates for EnterpriseOne 9.1</td>
</tr>
</tbody>
</table>
Executive Summary

IBM and Oracle have worked together to create this document for JD Edwards EnterpriseOne and IBM Power™ Systems server customers running IBM i. This document outlines key JD Edwards EnterpriseOne parameters and settings for optimal performance and is based on lab tests and customer feedback.

This white paper provides tuning techniques to optimize the performance of JD Edwards EnterpriseOne 9.0 and 9.1 with WebSphere® 7.0 and 8.5 on Power Systems servers with IBM i. JD Edwards EnterpriseOne is a comprehensive software package that provides Enterprise Resource Planning (ERP), Supply Chain Management (SCM), and Customer Relationship Management (CRM) capabilities using HTML-based user interfaces in a J2EE web server architecture.

These tuning recommendations cover general Power Systems server with i performance and the JD Edwards EnterpriseOne application. All components discussed are running on one or more IBM i partitions.

Performance Management for JD Edwards EnterpriseOne

One crucial factor in achieving optimal system performance is sufficient system resources implemented in a balanced configuration. An optimal system has sufficient processor, memory, disk, and network capacity to handle the application workload so that none of these factors is a performance bottleneck. The process of performance management which addresses allocation and balancing of resources is an ongoing, dynamic process. It consists of three key steps:

1. Measure and analyze
2. Adjust
3. Re-measure

For best system performance, these steps should be viewed as a process that is repeated on a regular basis. The results of these tuning iterations are used to resolve specific performance problems as well as to predict and schedule required system upgrades. Before tuning begins, an environment needs to meet Oracle’s Minimum Technical Requirements (MTRs). The MTR website is listed in Appendix B.

The following sections focus on examining and optimizing the performance of JD Edwards EnterpriseOne 9.x with tools release 8.98.x on IBM i 7.1 with IBM WebSphere Application Server 7.0 and 8.5. The information contained in this document originates from lab configurations and test scenarios executed by a joint team of IBM’s and Oracle’s JD Edwards development and technical professionals. The document also contains guidance based on customer feedback received by Oracle and IBM through a variety of channels. The intention of this document is to provide a better starting point from the default settings and MTR requirements for achieving a more efficient and tuned JD Edwards EnterpriseOne architecture. It is then necessary to adjust because every environment is different and actual results will vary.

Tuning the JD Edwards EnterpriseOne web server environment on the IBM Power Systems platform with i involves operating system components and the JD Edwards EnterpriseOne software package working together. The principles discussed here may also be applied to customized application software which
customers build on top of the basic software package. Customizations are beyond the scope of this document.

**General Power Systems server with i Performance**

General system performance can be managed using just a few IBM i commands and interfaces. The first is the Work with System Status (WRKSYSSTS) command which provides an effective method of monitoring and managing system resource utilization and job activity. The WRKSYSSTS screen provides high level feedback on the CPU utilization of the IBM i partition. A low CPU utilization indicates either that the system is not CPU constrained or that some other system or tuning limit is preventing the processor from being fully utilized. A high utilization is not necessarily a concern because most environments include low priority work which will use any processor resources that remain available after higher priority work is satisfied. The concern occurs only when end user response time and throughput become an issue for acceptable business activity.

For each memory pool, the WRKSYSSTS screen shows the faulting rate for database and non-database objects. Faulting occurs when a program references an object not currently in memory. It does not include planned I/O operations when a program explicitly requests a disk I/O. For example, DB2® support may pre-fetch database data into memory. This is counted as an I/O operation by other tools but it is not counted as a fault. Acceptable faulting rates depend on the workload, the number of disks, the disk adapters being used, etc. Definitive faulting guidelines are beyond the scope of this document but some useful links are included in Appendix B. The general guideline is that faulting rates should be low and memory should be added to pools with high faulting rates by shifting memory from pools with low faulting rates.

WRKSYSSTS also provides data at the system level including total auxiliary storage (disk) used, the total number of jobs in the system, and both the current and maximum amount of unprotected storage in use. These last two values are often the source of confusion. Unprotected in this context corresponds to temporary storage which is “unprotected” in that it is not necessarily protected if the system fails. The Single Level Store architecture support provided by IBM i allows the amount of temporary storage to exceed the total size of system memory. On most operating systems, this would be a source of grave concern while for IBM i, it is considered part of normal operations. Although it is recommended that these values be monitored, they do not provide useful data on memory usage.

The WRKSYSSTS display also shows activity levels and job transition statistics. (Depending on the assistance level being used for the command, press the F11 key to see job transition data.) The activity level (Max Active) indicates the maximum number of processes or threads that may be active in the memory pool at the same time. A low activity level value may result in additional delays when a process or thread is required to wait for a free activity level. This is indicated by job transitions from Wait to Ineligible or Active to Ineligible. A general rule of thumb is to increase the activity level value to avoid these transitions to ineligible.

The QPFRADJ system value can be used to automatically monitor and optimize memory pool sizes and activity levels. It can be enabled to optimize these values at IPL time, during normal run time, or both. It will move memory from one pool to another to reduce faulting rates in the most active pools. It will also increase the activity level to reduce or avoid job transitions to ineligible. The Work with Shared Pools (WRKSHRPOOL) command can be used to provide the system with additional guidance on this
automatic tuning; for example, minimum and maximum sizes for memory pool sizes can be specified. The display also includes the Paging Option for each pool which can be set to *FIXED or *CALC. *FIXED is the default and indicates that the system will not dynamically adjust the paging characteristics of the storage pool for optimum performance. The value *CALC enables dynamic adjustment of the paging characteristics and is generally recommended for all memory pools.

The Work with Disk Status (WRKDSKSTS) command can be used to monitor disk I/O activity including disk usage (percent full), disk utilization (percent busy), and detailed information on the number, size, and type of disk I/O operations. Performance can be strongly affected by disk utilization; the general guideline is not to exceed 40% busy.

System i Navigator Collection Services provide another way to gather performance data on the system and facilitate analysis and comparisons. It gathers all the data described above plus extensive information on jobs, resource utilization, seize and lock contention, journal performance, and so on.

All the commands, interfaces, and system values referenced in this section on general tuning are described in IBM System i and IBM i Information Center. The URL is listed in Appendix B.

**JD Edwards EnterpriseOne Application Tuning**

JD Edwards EnterpriseOne provides a complete suite of modular, integrated industry-specific business applications. JD Edwards EnterpriseOne is flexible, permitting the database, application logic, and user interface code to run entirely on one system or on separate systems.

**Measure and Analyze**

Beginning with toolset 8.97 Oracle introduced Server Manager for JD Edwards EnterpriseOne. This built-in tool is a web-based application used to manage the complete lifecycle of JD Edwards EnterpriseOne server products. Server Manager is the preferred method to manage, configure and monitor the different servers. It caches changes, provides dropdowns with valid values, maintains an audit history of changes, and allows comparisons between two different servers. All of the settings which follow are managed using this graphical interface and should no longer be set by editing the JDE.INI or JAS.INI files as in previous versions of JD Edwards EnterpriseOne.

Figure 1 shows the Server Manager kernel definitions panel, the starting point for these settings. To navigate use the following path: Access Server Manager (which displays the Management Dashboard) > Click on the Managed Home Location for the desired environment > Click on the desired EnterpriseOne Enterprise Server instance > Under Configuration on the left, click Kernel Definitions. Note that either the scroll bar on the right or the links at the top of the page permit navigation to the specific kernel definitions.
The JD Edwards EnterpriseOne enterprise server makes use of various jobs known as “kernels” to accomplish work for a web user request. Call object kernels, the primary process for processing user interface requests, run business functions (C-based business logic and database queries) on behalf of web users. The configuration interface provides two types of settings related to these jobs. The first setting, maximum number of processes, is the maximum number of jobs allowed to run. The second setting, auto-start process count, is the number of jobs automatically prestarted when JD Edwards EnterpriseOne services are initiated. Additional jobs are started if a request arrives and all jobs are currently active—but only up to the maximum number of jobs allowed.

Automatically starting a reasonable number of call object kernels is recommended but depends on the environment. In an environment where workload is high and users all log in at the same time, it may be reasonable to set the value for AutoStartProcess equal to the value for maxNumberOfProcesses. In an environment where workload is low and work does not all start at the same time, setting the value for AutoStartProcess equal to half of the value for maxNumberOfProcesses may yield better results.

Call object kernel jobs are capable of handling the requests for multiple users. Each job request caches information and uses system resources. Proper configuration becomes important because starting too
many jobs may cause excessive memory and CPU overhead. However, the presence of too few call object kernels could limit resources for each user thereby causing users’ response times to increase. Previous versions of this document indicated 15-20 users per call object kernel is optimal but feedback from customers indicated 6-10 users per call object kernel is more suitable. Lab tests yielded acceptable, comparable performance across the whole range between 8 and 20 users per kernel.

One way to establish the optimal number of users per kernel for a specific environment is to vary the call object kernel to user ratio and use Server Manager to monitor and compare the results. For example, Figure 2 displays the Call Object (BSFN) Stats screen which is sorted on the Average Time column. This field shows the average response times of the business function calls. The number of call object kernels which minimizes these response times during peak workload is the correct value for the environment.

To navigate use the following path: Access Server Manager (which displays the Management Dashboard) > Click on the Managed Home Location for the desired environment > Click on the desired EnterpriseOne HTML Server instance > Under Runtime Metrics on the left, click CallObject Stats, and sort Average time in descending order to see the panel shown in Figure 2.

Figure 2 - Call Object (BSFNs) Stats

There are two additional factors to consider when using the average time value to determine the optimal number of call object kernels. First, this approach can place too much emphasis on long running business functions, especially when there are a few that run much longer than most of the others. Second, this approach can also exhibit the law of diminishing returns. Once the number of call object kernels is
sufficient for good performance, little improvement will be gained from increasing their number. Very often, there is a fairly wide range of values that provide similar end user performance.

The two settings to change the number of call object kernels are shown in Figure 3. To navigate use the following path: Access Server Manager (which displays the Management Dashboard) > Click on the Managed Home Location for the desired environment > Click on the desired EnterpriseOne Enterprise Server instance > Under Configuration on the left, click Kernel Definitions > Click Call Object Kernel on the top.

Substitute the maximum number of call object kernels and auto-started call object kernels for the “enter value here” based on the number of users and workload characteristics. For example, if there are 200 users, and the decision is to have 8 users per kernel, all of them auto-started, both values would be 25 processes (kernels). Then evaluate performance as shown in Figure 2. Then try a second iteration perhaps using 15 users per kernel with half of them auto-started. If there are 200 users, the first value would be 14, and the second would be 7. Re-evaluate and compare the average response times of the business function calls. And so on, until the best performance is achieved.

The left-hand icon “i” is help text (which indicates the default setting) and the right-hand icon is history.

Security kernels authenticate the credentials for web users. Starting an optimal number of jobs is recommended for security kernels as well. One security kernel for every 100 users (minimum of two) is reasonable. Set the security kernel settings similarly as indicated in Figure 4 where “enter value here” is replaced by the values, such as 2 for 200 users and 2 for auto-start.

IBM Power Systems with IBM i Performance and Tuning Tips for Oracle’s JD Edwards EnterpriseOne 9.0 and 9.1
http://www.ibm.com/support/techdocs
© Copyright 2013, IBM Corporation
The network listener jobs, commonly referred to as jdenet_n jobs, receive JDENET based communications from web users and other clients and servers, and dispatch the messages to the appropriate kernel processes. For network jobs, one network job for every 20 call object kernels (minimum of two) is a reasonable setting. Set the maximum network processes as described above. For example, for up to 40 call object kernels it would be set to 2.

To navigate to the panel for kernel definitions shown in Figure 5, use the following path: Access Server Manager (which displays the Management Dashboard) > Click on the Managed Home Location for the desired environment > Click on the desired EnterpriseOne Enterprise Server instance > Under Configuration on the left, click Network and Queue Settings.

![Figure 5 - Network kernel settings](image)

1) Multi-threaded kernels in JDE.INI - thread settings and call object kernels

Beginning with 8.96_A1 system code, Oracle introduced the capability of multi-threading within call object kernels. With 8.98 system code, multi-threading is the default. Previously, call object kernels were single-threaded which meant that each kernel handled only one request at a time. Requests from other users assigned to the same kernel would wait in a queue to be processed in a round-robin fashion. In some environments under heavy workload, these requests had the potential of backing up behind long-running processes and thus negatively impacting response times.

The multi-threading capability addresses this problem of long-running business functions (LRBFs). Multi-threading guarantees that a business function call will be processed immediately. Function call requests are never queued when multi-threading is enabled. The three multi-threading parameters for call object kernels are:

- **singleThreadedMode** – This parameter will enable/disable multi-threading. A value of "Y" enables single-threading, and a value of "N" enables multi-threading (default).

- **ThreadPoolSize** – Default is 20. Every kernel maintains its own thread pool of worker threads. This value establishes the number of threads maintained in each pool. As the number of concurrent requests
increases in a kernel, the active threads for that kernel will grow. This will continue until the number of threads reaches the ThreadPoolSize value. This value defines the thread pool size, not the maximum number of threads that can be created within each kernel. The maximum number of threads can exceed ThreadPoolSize. In the case where the number of concurrent requests exceeds ThreadPoolSize, an additional thread will be created to accommodate the request. This ‘overflow’ thread will not become part of the pool. It will be destroyed immediately after the request for which it was created has been processed. Be aware that there is overhead associated with the creation and destruction of overflow threads. Choose values for both kernels and thread pool size that will minimize the need to create and destroy overflow threads. Entries will be written to the call object kernel logs when thread requirements exceed the ThreadPoolSize. A single message is entered in the log for every 1,000 threads created and destroyed to avoid excessive logging when thread pool size is exceeded. If this message appears, re-evaluate the configuration.

ThreadPoolSizeIncrement – This value defines the number of worker threads to be initiated at start-up (after the initial or main thread). It also defines the number of threads added to the pool when a new function call request arrives and no idle threads are available which will continue until the thread pool size has been reached. Default is 5.

Figure 6 shows the Server Manager panel associated with the multithreaded call object kernel settings with the pull down to specify the Thread Mode for Call Object Kernels followed by fields for the Thread Pool Size and Thread Pool Increment. This is the same panel as Figure 3.

Figure 6 - Multithreaded call object kernel settings

Internal testing with multi-threading enabled on Power Systems servers with i achieved the best performance with a thread pool size of 20 per call object kernel and an increment of 1. Thus, when enabling multi-threading, the recommended starting point is to set the thread pool size to 20 and the thread pool increment to 1. Based on discussion with Oracle experts, the ThreadPoolSize value should be set to twice the number of users per call object kernel (with a minimum or 20), so if the decision is to have 15 users per call object kernel, then the thread pool size should be increased to 30, with an increment of 1. Either recommendation is acceptable to follow as long as the overflow threads are never invoked.
There are a number of ways to monitor thread usage either through Server Manager, or through the operating system via iSeries Navigator or WRKACTJOB from the IBM i command line. See Figure 7 through Figure 9 for examples of each method. Also, check the call object kernels’ JDE.LOGs to make sure that the overflow message never occurs. The purpose of monitoring thread usage is to make sure that the ThreadPoolSize value is large enough.

To navigate to the Server Manager screen showing thread usage as in Figure 7, use the following path:
Access Server Manager (which displays the Management Dashboard) > Click on the Managed Home Location for the desired environment > Click on the desired EnterpriseOne Enterprise Server instance > Under Runtime Metrics on the left, click Process Detail, then click on the hyperlink of one of the Call object kernels and page down.

**Figure 7 - Thread detail screen for call object kernels**

Note: All JD Edwards EnterpriseOne shipped code is “thread-safe”, meaning that it can be run in a threaded environment without unpredictable interaction between the threads. Prior to enabling multi-threading, it is crucial to ensure that any customized Business Function code is rendered “thread-safe”. The main issue is that any shared objects must be accessed in a synchronized manner. See the Oracle Redpaper “Multi-threaded Kernels” listed in Appendix B for more information.

**Figure 8 - Using iSeries Navigator for thread count in call object kernels**

Note: The Metadata kernel which is [DEF30] in the JDE.INI helps manage access to the XML spec format, which has replaced TAM specs. Since Metadata is initiated as a JVM process, it normally has a higher thread count than call object kernels. Refer to Oracle Redpaper “Platform Configuration”
Considerations for XML Metadata for JD Edwards EnterpriseOne” listed in Appendix B for information on the XML Metadata kernel.

**Figure 9 - WRKACTJOB and F11 twice for number of threads in call object kernels**

2) Single-threaded kernels in JDE.INI - number of call object kernels

When call object kernels are in single-threaded mode, monitor the Server Manager system summary and, if outstanding requests are backing up, either increase the number of call object kernels or consider switching to multi-threading. To navigate to the system summary use the following path: Access Server Manager (which displays the Management Dashboard) > Click on the Managed Home Location for the desired environment > Click on the desired EnterpriseOne Enterprise Server instance > Under Runtime Metrics on the left, click Process Detail.

The recommendation for security kernels, network jobs, and numberAutoStartProcess settings remains the same as multi-threaded. See the preceding section for multi-threaded mode.

3) Maximum users for the JAS server in JAS.INI

The maximum users setting in the JAS.INI acts as a threshold. Once the maximum number of users has been reached, no additional users are allowed to log in to JD Edwards EnterpriseOne until another active user logs out. Set the MAXUser value equal to the maximum number of users to be concurrently connected to the JAS server by replacing the “enter value here” as shown in Figure 10. To navigate to...
this panel, use the following path: Access Server Manager (which displays the Management Dashboard) > Click on the Managed Home Location for the desired environment > Click on the desired EnterpriseOne HTML Server instance > Under Configuration, click Web Runtime.

![Web Runtime](image)

**Figure 10 - Max Users settings**

4) **JDENET connections for the JAS server in JAS.INI**

The connections between the JAS server and the enterprise server are TCP/IP-based and are referred to as JDENET connections. The number of JDENET connections should be the same as the number of users accessing the JAS server. To increase the number of connections, navigate to the pool settings screen and replace “enter value here” as shown in Figure 11. Use the following path: Access Server Manager (which displays the Management Dashboard) > Click on the Managed Home Location for the desired environment > Click on the desired EnterpriseOne HTML Server instance > Under Configuration on the left, click Network Settings.

![JDENET Configuration](image)

**Figure 11 - Pool Size settings**
JDENET connections screen in Server Manager can be used to verify the setting. During peak activity, monitor the "threads waiting for socket" column. This number should usually be "0" to ensure that no latency problems exist for JDENET connections.

To navigate to this panel, use the following path: Access Server Manager (which displays the Management Dashboard) > Click on the Managed Home Location for the desired environment > Click on the desired EnterpriseOne HTML Server instance > Under Runtime Metrics on the left, click JDENET Stats.

![Figure 12 - Monitor JDENET Connection Pool settings](image)

4) Optimize the use of SQL Packages

SQL packages are i5/OS objects that contain the control structures and the access plan data necessary to process SQL statements when running a distributed program, such as JD Edwards EnterpriseOne. Because SQL packages are shared resources, the information built when a statement is prepared is available to all the users of the package. This saves time, especially in a JD Edwards EnterpriseOne environment where many of the users and processes execute the same or similar queries.
All the various types of database interfaces used by JD Edwards EnterpriseOne make use of SQL packages. There are two important considerations when using SQL packages in a JD Edwards EnterpriseOne environment:

- Ensuring SQL packages are being used effectively
- Knowing when to delete SQL packages
  - When underlying metadata has changed in the SQL package
  - When the SQL package will no longer be used (to clean up disk space)

JD Edwards EnterpriseOne has a setting called SQL Package Library which controls package reuse for UBEs. The possible values are as follows:

- The value 2, the default setting, is designed to handle a special kind of UBE called Table Conversions (TCs) which runs during an upgrade. It means that SQL packages for UBEs will never be reused. This is necessary because TCs all share the same name. Turning off their reuse improves performance of upgrades as the next UBE would not find the correct framework if it attempts to use the SQL package created by the previous UBE. A setting of 2 will result in the creation of SQL package in QRECOVERY starting with the letter “T” followed by the job number. Leaving this value at 2 for normal operations is not recommended because UBEs will run slower as they will not be able to reuse the SQL package information.
- A value of 0 or 1 allows reuse of SQL packages for UBEs and is better for performance for normal operations. If the value is 0, the SQL package will be created in the library specified in JDE.INI [INSTALL] under DefaultSystem. If the value is 1, the SQL package will be created in library QRECOVERY.

Changing the SQL Package Library setting to a 1 or 0 is highly recommended during normal operations in order to take advantage of the benefits of SQL packages for UBEs. This is especially important in environments that have many short running UBEs. When the SQL packages aren’t shared, it will result in the creation of an excessive number of SQL package and slower throughput for short running UBEs. When SQL packages are not reused, non-database page fault rates are increased and memory consumption may be adversely impacted.

To change this from Server Manager use the following path: Access Server Manager (which displays the Management Dashboard) > Click on the Managed Home Location for the desired environment > Click on the desired EnterpriseOne Enterprise Server Instance > Under Configuration on the left, click Database Configuration.
Figure 13 - Database Configuration Panel

Change the value of SQL Package Library from 2 to 1. Click Apply.

Note that the setting does not affect the SQL packages created for kernel jobs where the naming convention begins with OW or OW_M (manual commit) followed by the kernel’s unique job number, for example, OW123456.

Packages must be deleted when the underlying metadata for statements stored in the package has been changed. If a table, view, stored procedure, or other SQL object is altered, information in the package may not be updated. Therefore, JD Edwards EnterpriseOne SQL packages should be deleted whenever significant changes have been made to the database, operating system, or hardware. Examples of significant changes are those that may cause a large amount of access plan rebuilds, such as applying a database group PTF, upgrading the operating system, or applying a JD Edwards EnterpriseOne Electronic Service Update (ESU) when it contains table changes. Although not required, it may also be useful to delete SQL packages when indexes are created or deleted. Deleting SQL packages can also be made a part of a routine maintenance schedule. Because any user SQL package that is needed is recreated when the JD Edwards EnterpriseOne application services are started, there is little functional risk in deleting them but it will degrade performance as statement and access plan data must be rebuilt.

As stated earlier, JD Edwards EnterpriseOne kernel jobs use SQL packages beginning with OW or OW_M created in the QRECOVERY library. Each time the enterprise server is ended and restarted, new SQL packages are built and associated with the new kernel jobs. Old SQL packages associated with kernel jobs that have ended are never used again and can be deleted to save disk space.
Many customers delete SQL packages based on concerns of reaching the maximum size which defaults to approximately 500 MB. If this is a concern for the environment, consider using the SQL_INCREASE_PKG_LIMIT value which allows them to grow to approximately 1 GB. The procedure for implementing this value is documented in IBM support document: http://www-01.ibm.com/support/docview.wss?uid=nas16abf236a1dd99bd68625778c00715590

5). Controlling the Priority and Pool of QSQSRVR jobs

The current JD Edwards EnterpriseOne tools release uses SQL Server Mode to process all database work. When SQL Server Mode is enabled, database requests are sent to a pool of prestart jobs named QSQSRVR. The QSQSRVR jobs pickup the requests in a one-to-one relationship with the requesting job. As a result, the JD Edwards EnterpriseOne jobs do not process their own database request. The corresponding QSQSRVR job performs all database requests. The default configuration on IBM i will run all the of the QSRSRVR prestart jobs in the QSYSWRK subsystem. Most of the time this configuration will provide acceptable performance.

However, there are several scenarios when it may not be optimal to have the QSQSRVR jobs run in the QSYSWRK subsystem.

- When you want to have better control of the memory pool for database activity. When the QSQSRVR jobs run in the same pool as their initiating job, you will have more control over the amount of memory that is used for specific processes. For example, you can help ensure that memory pages used by interactive jobs are not paged out due to batch activity.
- When you want the QSQSRVR jobs to run at the same priority as the job that initiated them. The jobs in the QSYSWRK subsystem run at priority 20. As a result, the database activity for UBEs could have some impact to your interactive workload. Additionally, if UBEs run under multiple priorities on the system, the database activity for the lower priority UBEs may impact the runtime of the higher priority UBEs.
- And, you may find it easier to troubleshoot or isolate heavy SQL processing when the QSQSRVR jobs runs in the same subsystem as the initiating job.

IBM has provided PTFs to allow you to have more control of the subsystem that runs the QSQSRVR jobs. These PTFs provide an environment variable which, when present, will run the QSQSRVR jobs in the same subsystem as the job that initiated them. On IBM i 6.1, you must have PTF SI33949 installed. The required PTF number for IBM i 7.1 is SI40585.

To enable the environment variable, use the following command:

ADDENVVAR ENVVAR(QIBM_SRVRMODE_SBS) VALUE("SAME") LEVEL(*SYS)

When using the environment variable for optimal performance, a prestart job entry should be created in each subsystem that uses the QSQSRVR jobs. For JD Edwards EnterpriseOne, the subsystems which need the prestart job entry are:

- JDEE900 in library E900SYS or JDEE910 in library E910SYS
- The WebSphere subsystem for WebSphere 7.0 is usually QWAS7 in library QWAS7 and the subsystem for WebSphere 8.5 is QWAS85 in library QWAS85.
- The batch subsystem(s). By default this is the QBATCH subsystem, you may have one or more batch subsystems in your environment.
When adding the prestart job entry, you should specify the number of QSQSRVR jobs to create when the subsystem starts. There are some general guidelines for setting the initial number of prestart jobs for Websphere, JDE and batch subsystems.

Each batch job will create between 10 and 12 database connections. In turn, each connection will use one prestart job. Therefore, the initial number of prestart jobs should be approximately equal to ten times the number of the concurrent active UBEs.

The number of database connections created in the WebSphere subsystem will be determined by the HTML server configuration. This is defined in the JDBJ database configuration section in Server Manager. To view this, use the following path: Open Server Manager > Click on the Managed Home Location for the desired environment > Click on the desired EnterpriseOne HTML Server instance > Under Configuration on the left, click JDBJ Database Configuration. Scroll to the JDBJ Connection Pools section,

![Figure 14 - JDBJ Connection Pools settings](image)

The initial connections setting will be number of database connections the JAS creates for each instance of the HTML server. To determine the number of connections being used, open Server Manager. Click on the Managed Home Location for the desired environment > Click on the desired EnterpriseOne HTML Server instance > Under Runtime Metrics on the left, click Database Connections. The JDBJ Pooled Database Connections section will show the total number of connections created by JAS at the current time.
Figure 15 - JDBJ Pooled Database Connections details

Set the number of prestart jobs for the QWAS7 subsystem, based on the number of JDBJ connections in use. Ensure that you consider all HTML Servers that are configured in all WebSphere instances on the server.

For the JDEE900 or JDEE910 subsystem, you can expect to have between seven and nine database connections per kernel.

After determining the initial number of prestart jobs to create, add the prestart job entry to each subsystem (JDEE900 or JDEE910, QWAS7 or QWAS85 and QBATCH) using the following set of commands for each subsystem:

- **ENDSBS** <subsystem-name>
- **ADDPJE** SBSD(<library>/<subsystem-description-name>)
  PGM(QSYS/QSQSRVR) STRJOBS(*YES) INLJOBS(x) THRESHOLD(y) ADLJOBS(z)
  - x - The number of initial jobs to be created
  - y - When the pool of QSQSRVR jobs is reduced below this number, more will be started
  - z - The number of additional jobs which will be created when the threshold has been reached.
- **STRSBS** <subsystem-name>

You must use the following command when performing these steps above on the JDEE900 or JDEE910 subsystem: **ADDPJE** SBSD(E900SYS/JDEE900) PGM(QSQSRVR) INLJOBS(x) THRESHOLD(y) ADLJOBS(z) CLS(E900SYS/JDENET) or **ADDPJE** SBSD(E910SYS/JDEE910) PGM(QSQSRVR) INLJOBS(x) THRESHOLD(y) ADLJOBS(z) CLS(E910SYS/JDENET), respectively.

The class definition for the JDEE900 or JDEE910 subsystem must be specified.

To ensure that no old QSQSRVR jobs remain in QSYSWRK, stop and restart all QSQSRVR jobs in the QSYSWRK subsystem using the following commands:

- **ENDPJ** SBS(QSYSWRK) PGM(QSQSRVR) OPTION(*IMMED)
- **STRPJ** SBS(QSYSWRK) PGM(QSQSRVR)

Note: These configuration changes should not be implemented on an active environment. They should be tested thoroughly in a test environment before applying in production.

Additional important things to consider before implementing this change are:

- The JDEE900 or JDEE910 subsystem is replaced when an update or service pack is applied. You must add the prestart job entry to the new subsystem description when a JD Edwards EnterpriseOne tool fix is applied.
If a prestart job entry for QSQRVR does not exist in a subsystem, the QSQSRVR job will utilize a Batch Immediate job (BCI). This can result in reduced performance for the jobs in that subsystem.

For subsystem QHTTPSVR, the environment variable has no effect. QSYSWRK will continue to be used for those SQL mode jobs.

The QSQSRVR jobs count toward the MAXJOBS parameter which is defined for the subsystem. If your subsystem description is configured with this parameter, you will need to make sure that the MAXJOBS is set sufficiently high to allow for all QSQSRVR jobs to process (see Figure 16). Alternatively, you can set the MAXJOBS parameter to *NOMAX, which is the default when subsystems are created.

You will need to monitor the activity levels in the pools where the three subsystems run to prevent wait to ineligible or active to ineligible transitions. Figure 17 shows an example where the activity level of a storage pool needs to be increased.
The JDEE900 or JDEE910 subsystem will take longer to start than usual. All prestart jobs are by default created when a subsystem is started. Therefore, all QSOSRVR jobs will be started before any JDE kernel processes become active.

For additional information on implementing these PTFs, see the references to the PTFs listed in Appendix B.

NOTE: The description for IBM i 6.1 PTF (SI33949) provides the most comprehensive explanation of the functionality and implementation of these PTFs.

6). Adding a Low Priority Job Queue

To take better advantage of system resources, maintain a balance between good response times and batch throughput, and to improve batch throughput for critical jobs, designate specific UBEs as low priority. To do this, create a new subsystem, job queue and class. The new class is defined with a run priority lower than the existing batch job queues. Specific UBEs are then configured to run in the new job queue.

Note: These configuration changes can be implemented on an active environment. It is, however, recommended that this be done in test environment first.

In this example, we create a subsystem named QBATCH60. This subsystem will use a class QBATCH60 which has a run priority of 60. We also create a job queue named QBATCH60 which will be single-threaded and will run in the QBATCH60 subsystem.

Create new subsystem

To create the subsystem, use the CRTSBSD command: This subsystem will run in pool *BASE.

CRTSBSD SBSD(QBATCH60) POOLS((1 *BASE *N))
### Create Subsystem Description (CRTSBSD)

**Type choices, press Enter.**

<table>
<thead>
<tr>
<th>Subsystem description</th>
<th>&gt; QBATCH60</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Library</td>
<td>&gt; QGPL</td>
<td>Name, *CURLIB</td>
</tr>
</tbody>
</table>

**Storage pools:**

- **Pool identifier**: > 1
  - **Storage size**: > *BASE 1-10
  - **Activity level**: Number

**Maximum jobs**: > *NOMAX 0-32000, *NOMAX

**Text 'description'**: > *BLANK

---

*Figure 18 - Create subsystem description command*

In the command above, we have created a one pool identifier, 1, which references the *BASE pool.

#### Create new class

To create the class, use the CRTDUPOBJ command to create an identical class to the QBATCH.

```sql
CRTDUPOBJ OBJ(QBATCH) FROMLIB(QSYS) OBJTYPE(*CLS) TOLIB(QGPL) NEWOBJ(QBATCH60)
```

*Figure 19 - Create Duplicate Object (CRTDUPOBJ)*

The class QBATCH60 will be created in library QGPL. By default the QBATCH class has a run priority of 50. Change this to priority 60 by using the CHGCLS command:

```sql
CHGCLS CLS(QGPL/QBATCH60) RUNPTY(60)
```

#### Create job queue

To create the new job queue QBATCH60, use the CRTJOBQ command:

```sql
CRTJOBQ JOBQ(QBATCH60)
```

**Modify subsystem**

Next, modify the subsystem to add the job queue and to create a routing entry that will use the QBATCH60 class. Add the job queue to the QBATCH60 subsystem by using the ADDJOBQE command:

```sql
ADDJOBQE SBSD(QBATCH60) JOBQ(QBATCH60) MAXACT(1)
```

The MAXACT parameter specifies that only one job will be active for this job queue at any time.
Add the routing entry to the subsystem QBATCH60, to specify that any job which runs in the system will use the QBATCH60 class. To do this use the ADDRTGE command:

```
ADDRTGE SBSD(QBATCH60) SEQNBR(9999) CMPVAL(*ANY) PGM(QCMD) CLS(QGPL/QBATCH60)
```

### Figure 20 - Add routing entry command

Start the subsystem using the STRSBS command:

```
STRSBS QBATCH60
```

### Add job queue to JD Edwards EnterpriseOne

Before you can use this new job queue, you must define the queue within JD Edwards EnterpriseOne. Sign-in to the JD Edwards EnterpriseOne HTML client and fast path to program P986130.

Click Add.

### Figure 21 - Job queue revisions

For host name, enter the name of the enterprise server where you created the job queue. Enter the queue name, QBATCH60. Specify the job queue status 01, (Active). The Queue Type is 02 (AS/400 Queue). Verify that the port number matches your enterprise server configuration.
Sending UBEs to new job queue

There are multiple ways that you can have specific UBEs use the new job queue. The method that you choose depends on how you typically execute the jobs which you want to run in the new job queue.

1. The first option you have is to create or modify specific versions which will use the job queue QBATCH60. Within the development tools, create a new version or edit an existing version and specify the new job queue.

Figure 22 - OMW version detail

2. If you use the JD Edwards EnterpriseOne job scheduler,(P93100) you can specify the job queue the UBE will use when you create the job schedule.

Figure 23 - Job scheduler batch overrides
NOTE: For more information about using the JD Edwards EnterpriseOne scheduler, see: http://download.oracle.com/docs/cd/E17984_01/doc.898/e14721/scheduler_application.htm

3. If using a CL program or CL command, specify the job queue in the RUNUBE command. The screen in Figure 24 shows the parameters for the RUNUBE command. You will specify QBATCH60 in the job queue parameter.

![Figure 24 - RUNUBE command](image)

4. The final option for submitting UBEs to the new job queue is to override the job queue when the job is submitted.
   a. From the Submit Job –Version Prompting screen, select Form->Advanced

![Figure 25 - Version prompting screen](image)
b. Select Override Job Queue, and Click OK.

Figure 26 - Advanced version prompting screen

Validating the setup

To validate that the setup is working as expected:

- Submit a UBE to job queue QBATCH60
- Enter the command WRKACTJOB SBS(QBATCH QBATCH60)
- Press F11
- Verify that the UBE you submitted is running under the subsystem QBATCH60
- Verify that the priority of the UBE is 60 as shown in Figure 27

Figure 27 - Work with active jobs screen
Appendix A – Advanced Performance Tuning

IBM iDoctor for IBM i

IBM iDoctor for IBM i is a suite of performance tools that can be used by the performance expert or novice to collect, investigate and analyze performance data on Power Systems servers running i. The tools are used to monitor overall system health at a high "overview" level or to drill down to the performance details within job(s), disk unit(s) and/or programs over data collected during performance situations.

The iDoctor family of products and services includes:

- IBM iDoctor for IBM i Job Watcher
- IBM iDoctor for IBM i Job Watcher – Collection Services Investigator
- IBM iDoctor for IBM i Job Watcher – Disk Watcher
- IBM iDoctor for IBM i PEX Analyzer
- IBM iDoctor for IBM i Heap Analyzer Heap Analysis Tools for Java

For additional information on IBM iDoctor for i, visit the following URL: https://www-912.ibm.com/i_dir/idoctor.nsf

Oracle’s Performance Workbench

Performance Workbench is an Oracle-created tool which parses and formats the content of JD Edwards EnterpriseOne debug log files and produces reports which contain timing and profiling data from JD Edwards EnterpriseOne code. It is widely used by customers, business partners, and Oracle internal developers. The debug logs encompass JD Edwards EnterpriseOne business functions, event rules, database calls and tables. Logging over extended periods of time can yield debug logs that are very large in size (>1GB) which are difficult to open by conventional NT tools like Notepad. Very large and otherwise unwieldy debug log files can be reduced to manageable statistical summaries by this tool which can be used to analyze and troubleshoot JD Edwards EnterpriseOne code. It is downloadable from Oracle as a zip file and runs on a Windows® system. Extensive help files with usage examples are included with the tool. For more information visit the URL (userid and password required): https://metalink3.oracle.com/od/faces/secure/km/DocumentDisplay.jspx?id=747328.1&h=Y
Appendix B – Additional Information

Contact information
Please send questions or comments via email to IBM Oracle International Competency Center at ibmoracle@us.ibm.com

References

General System Tuning
- IBM eServer iSeries Performance Management Tools (03 October 2005)
- IBM Power Systems Performance Capabilities Reference IBM i operating system Version 6.1
  http://www-03.ibm.com/systems/resources/pcrm2804.pdf
- Performance Management for IBM System i Home Page
  http://www-03.ibm.com/servers/reserver/iseries/perfmgmt/
- IBM System i and IBM i Information Center
  http://publib.boulder.ibm.com/infocenter/iseries/v6r1m0/index.jsp
- Oracle Minimum Technical Requirements (MTRs) (userid and password required)
  https://metalink3.oracle.com/od/faces/secure/km/DocumentDisplay.jspx?id=747323.1
- IBM / Oracle Informational APARs
  http://www-03.ibm.com/systems/i/advantages/oracle/
- IBM Preventative System Planning for System i - PSP

Advanced Performance Tools
- IBM iDoctor for IBM i website
  https://www-912.ibm.com/i_dir/idoc.nsf
- IBM iDoctor iSeries Job Watcher: Advanced Performance Tool (11 March 2005)
- Oracle’s Performance Workbench
  https://metalink3.oracle.com/od/faces/secure/km/DocumentDisplay.jspx?id=747328.1&h=Y

JD Edwards EnterpriseOne
- JD Edwards EnterpriseOne documentation
  http://www.oracle.com/technology/documentation/jde.html
- Server Manager for JD Edwards EnterpriseOne Guide (user id and password required)
**Oracle Redpapers (userid and password required)**

- Multithreaded Kernels: Preparing for This Throughput Enhancement and Best Practices Business Function Coding (September 2005) Doc ID 748330.1
  
  ![Image](https://support.oracle.com/CSP/ui/flash.html?tab=KBHome(page=KBHome&id=(),(page=KBNavigator&id=(bmDocType=WHITE%20PAPER&bmDocTitle=Multithreaded%20Kernels&viewingMode=1143&from=BOOKMARK&bmDocID=748330.1&bmDocDsrc=DOCUMENT))

- Platform Configuration Considerations for XML Metadata for JD Edwards EnterpriseOne (January 2006) Doc ID 748356.1
  

- Considerations for SQL Server Mode Database Connections Doc (November 2010) ID 841314.1
  
  ![Image](https://support.oracle.com/CSP/ui/flash.html?tab=KBHome%28page=KBHome&id=%28%29%29,(page=KBNavigator&id=%28bmDocType=HOWTO&bmDocTitle=E1:%20OS:%20Considerations%20for%20SQL%20Server%20Mode%20Database%20Connections&viewingMode=1143&from=BOOKMARK&bmDocID=841314.1&bmDocDsrc=KB%29%29)

**Additional Resources**

- For additional copies of this document and for other white papers please visit http://www-03.ibm.com/support/techdocs/atsmastr.nsf/Web/TechDocs

- IBM Power Systems with IBM i performance and tuning tips for Oracle’s JD Edwards EnterpriseOne WebSphere-based HTML servers please visit: http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP101777

- For additional IBM Redbooks not listed above, please visit http://ibm.com/redbooks

- For specific information on JD Edwards Solutions from Oracle on System i, please visit http://www-03.ibm.com/servers/eserver/iseries/solutions/oracle/index.html

- For additional information on integrated, collaborative enterprise solutions from IBM and Oracle’s JD Edwards EnterpriseOne, please visit http://ibm.com/solutions/businesssolutions/oracle

- For additional information on the PTFs to configure the QSQSRVR jobs, please visit:

  - For V6R1:
    
    ![Image](http://www-01.ibm.com/support/docview.wss?uid=nas3948c6256188c186586257528005d7abb)

  - For V7R1:
    
    ![Image](http://www-912.ibm.com/a_dir/as4ptf.NSF/b3cb9d42f672b70f86256739004afa0f/3cc2a34f66659fbe86257775006ed861?OpenDocument)
Trademarks and special notices

© Copyright. IBM Corporation 2012. All rights reserved.

References in this document to IBM products or services do not imply that IBM intends to make them available in every country.

IBM, the IBM logo, DB2, Power Systems, and WebSphere are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:
Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
Microsoft, Windows, Windows Server, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
Other company, product, or service names may be trademarks or service marks of others.
The information provided in this document is distributed “AS IS” without any warranty, either express or implied.
The information in this document may include technical inaccuracies or typographical errors.
All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer.
Information concerning non-IBM products was obtained from a supplier of these products, published announcement material, or other publicly available sources and does not constitute an endorsement of such products by IBM. Sources for non-IBM list prices and performance numbers are taken from publicly available information, including vendor announcements and vendor worldwide homepages. IBM has not tested these products and cannot confirm the accuracy of performance, capability, or any other claims related to non-IBM products. Questions on the capability of non-IBM products should be addressed to the supplier of those products.
All statements regarding IBM future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. Contact your local IBM office or IBM authorized reseller for the full text of the specific Statement of Direction.
Some information addresses anticipated future capabilities. Such information is not intended as a definitive statement of a commitment to specific levels of performance, function or delivery schedules with respect to any future products. Such commitments are only made in IBM product announcements. The information is presented here to communicate IBM’s current investment and development activities as a good faith effort to help with our customers’ future planning.
Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.
Photographs shown are of engineering prototypes. Changes may be incorporated in production models. Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.