IBM / J.D. Edwards International Competency Center
January 2003

WEB CLIENT TUNING TIPS FOR iSERIES

V5R2, WebSphere 4.0, HTTP Server (powered by Apache), J.D. Edwards 5 ERP 8.0 / OneWorld Xe, SP20 & SP21

By Gerrie Fisk
iSeries I/T Specialist
IBM / J.D. Edwards International Competency Center

IBM | e business solutions | JDEDWARDS

Table of contents

Introduction

For the best performance of the web client on the iSeries™ server, parameters in the WebSphere® Application Server, the HTTP Server, and the J.D. Edwards® JAS and enterprise servers can be tuned. Many of these configurable parameters are based on the number of users. This guide documents the settings on the iSeries server that we found during testing to provide the best overall end-user response time, while keeping the system load as low as possible.

For background or additional information, please refer to the ERP 8.0 (or OneWorld®) Web Server Installation Guide, and the IBM website: http://www1.ibm.com/servers/eserver/iseries/software/websphere/wsappserver/indexAE40.html

Consider these guidelines as a starting point and please read the important disclaimer information in Appendix A to understand the limitations of this document. Your environment will be different from our lab and your results may vary. For example, if you also have fat or TSE clients, a different mix of batch jobs (UBEs), or other applications running on your system, additional iterations may be required to optimize the performance of your implementation. Keep in mind that there are steady improvements and fixes in the iSeries server software and ERP 8.0 / OneWorld code so, over time, the guidelines established by our ongoing testing will continue to improve as well.

Test environment

We tested web clients with V5R2 of OS/400® on the iSeries server with WebSphere 4.0.4 and OneWorld Xe SP20_B1.

We used two different hardware configurations. The first configuration, All-in-One (AIO), had everything installed on a single iSeries server: OneWorld Java™ server, WebSphere, HTTP server (powered by Apache), OneWorld application and database servers. We ran the AIO configuration with an SP21-version JDBC driver because of faster performance over SP20. We recommend SP21, therefore, for the AIO configuration. The second hardware configuration that we tested,

Highlights

Virtual 3-Tier (V3T), had one iSeries server set up as the OneWorld Java™ server with WebSphere and HTTP server (powered by Apache), and another iSeries server set up as the OneWorld enterprise (application and database) server. See diagrams of our two hardware configurations in figures 2 and 3.

The software levels tested in figure 1 were current at the time of our measurements and are included only as background information. For the latest IBM PTF recommendations check Informational APAR II13384 for V5R2:
http://ibm.com/eserver/iseries/developer/solutionservers/jde/
This website includes Informational APARs for other releases as well as other documentation pertinent to running J.D. Edwards software on the iSeries server.

Software for test environment (WRKPTFGRP):	Description:	Service Pack:
OS/400	V5R2	Cum C2260520
Database	SF99502	Service Pack #2
OneWorld	Xe	SP20_B1*
HTTP	SF99098	Service Pack #4
WebSphere Advanced 4.0.4	SF99148	Service Pack #2
Java™	SF99169	Service Pack #3
iSeries Express V5R2 (only needed on fat client to do the Java™ generation, but not required for ERP 8.0 / OneWorld HTML clients)	SI05853	Service Pack #2

Figure 1. iSeries Testing Environment

** To test the AIO configuration we included in our tests a JDBC driver which is functionally equivalent to SP21. This functionality is available to customers in SP21.*

Highlights

Lab tests included two different hardware configurations, All-in-One (AIO), and Virtual 3-Tier (V3T).

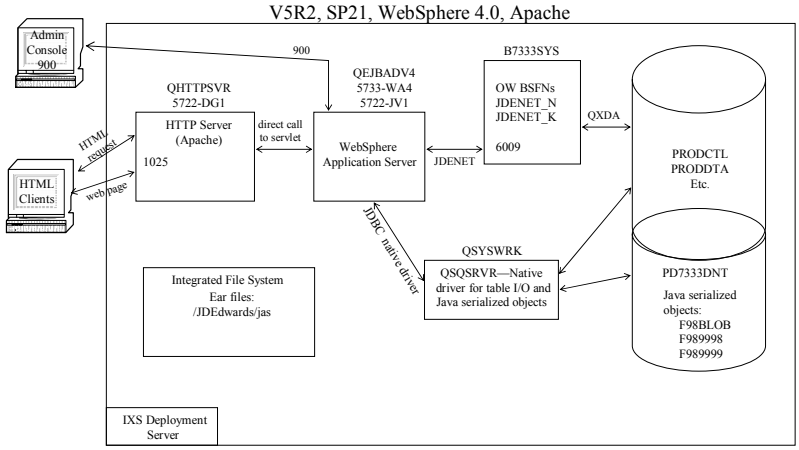


Figure 2. "All In One" (AIO) Configuration

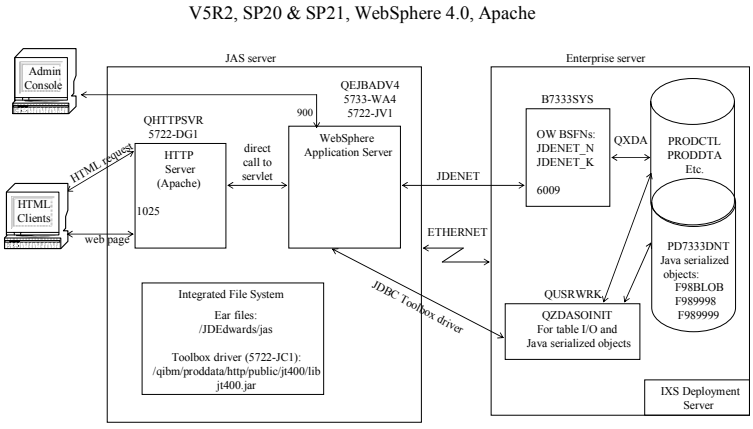


Figure 3. "Virtual 3 Tier" (V3T) Configuration

Highlights

Each JVM can now run as many as 1500 users.

WebSphere application server

Most of these settings are accomplished by starting the WebSphere Administrative Console on your PC. Expand Nodes, expand your node, expand Application Servers, and click on your application server. Make the changes as directed and then click Apply, click OK, and watch for the console message below that says 'Command "JVMx.ModfiyAttributes" completed successfully'. Stop and restart your server for these changes to take effect.

1. Users per WebSphere application server instance (JVM): Web clients use a browser running Internet Explorer or Netscape which connects through the HTTP Server running on the iSeries server to pass requests to WebSphere (5733-WA4) running ERP 8.0 / One-World Java™ server. Each application server instance is called a JVM, because each instance runs in its own Java™ Virtual Machine. In the lab, we ran 1500 users in an instance and, for best performance, we recommend that each JVM supports up to 1500 users. This maximum user recommendation per JVM has increased due to WebSphere 4.0, J. D. Edwards Java™ server service pack improvements, and tuning.

Note: Having more than 1500 users in a JVM increases the memory contention within WebSphere and therefore elongates the response time for these web clients. If you do have more than 1500 users, you should create an additional instance by configuring a clone. See the ERP 8.0 (or OneWorld) Web Server Installation Guide for instructions.

2. Heap size and subsystem memory settings: For the heap size, set an initial heap of 2.5 MB memory per user and no maximum heap. WebSphere's subsystem needs 8 MB memory per user.

You can control the amount of the iSeries memory available for the Java™ server by varying the heap size. WebSphere on the iSeries server performed best when using only an initial heap size setting and allowing the system to grow the heap to the maximum size needed. The iSeries server is unique in managing the maximum heap size; both AIX® and Windows® implementations require the maximum parameter

Highlights

to be set. Go to JVM settings tab. Enter the initial heap size you have selected and ensure that maximum heap size is blank.

Note: The lower your initial heap, the more CPU will be used for the extra garbage collections; the higher amount of memory in the initial heap, the less CPU used. So if your CPU is constrained, you could adjust the initial heap higher, up to 7.5 MB memory per user.

Setting the initial heap size for your user count is a key factor in performance.

You can review your heap size, while running your normal workload, by using J. D. Edwards "web" SAW and checking the System Summary page for the amount listed under "Heap memory allocated by VM". At first, this number will be equal to the initial heap and some overhead for WebSphere. As users are added, it increases with user work and decreases with garbage collections. In our tests, we found total heap memory utilization to be somewhat less than 3 times the initial setting. We recommend allocating 8 MB memory per user (which includes overhead for WebSphere) in the subsystem where WebSphere runs, for optimal performance.

3. Garbage collection setting: Use the –noclassgc setting. Java™ manages its memory usage by periodically cleaning up unused memory, which is referred to as garbage collection. WebSphere loads the ERP 8.0 / OneWorld Java™ classes and, by default, garbage collects them as needed. Because these classes are constantly used in this environment, we set the command line arguments to prevent the classes from being garbage collected by using the setting –noclassgc.

Go to the JVM settings tab, click on Advanced JVM settings, and add the -noclassgc argument to the command line arguments, if it is not there already.

Extra tip: Add the following to the command line argument -Dos400.user.timezone=MST, where MST is Mountain Standard Time. Change this according to your location. Although it does not affect performance, this setting makes the time stamps in your standard out and standard error logs correct.

Highlights

Web Container Service settings control threads and keep alives.

4. Web Container Service: Each client uses a JDBC connection that occupies a thread within the JVM. WebSphere also utilizes additional threads as needed for the QSQSRVR and/or QZDASOINIT jobs.

a. Thread control: Change the minimum number of threads (default 25) to equal the number of users you are expecting during normal operations, and set the maximum number of threads (default 50) to equal the highest number of users you ever expect plus one (max of 1501).

Note: Setting the maximum number of threads below your actual user count would force users to wait for a thread, increasing their response times but decreasing CPU utilization. If your system becomes CPU constrained, you might choose to lower the maximum setting to conserve CPU at the expense of longer response times. Remember to reset this value after a system upgrade.

Go to the Services tab, then choose Web Container Service and Edit Properties, and under the General tab, change minimum and maximum thread size.

b. Keep alives: WebSphere initially sets the default for max keep alive threads to 45, which is 90% of default maximum threads of 50. If you have changed the maximum threads in the previous section, you need to increase this number to be 90% of the new maximum threads value. This sets the maximum concurrent keep alive (persistent) connections in the WebSphere container pool.

Go to the Services tab, then choose Web Container Service and Edit Properties, and under the Transport tab, change max keep alive connections to equal 90% of your maximum number of threads allowed.

5. Java™ program optimization: You do not need to create Java™ programs for the ERP 8.0 / OneWorld Xe .class files that are delivered with JDE's web client, because WebSphere does not use these Java™

Highlights

programs even if they are created. It only uses the JIT compiler. (This is the default, so no action is required.)

HTTP Server (powered by Apache)

The IBM HTTP Server for iSeries (5722-DG1) offers two HTTP servers on the iSeries server. For WebSphere 4.0, we recommend the HTTP server (powered by Apache).

Although you can edit the HTTP configuration file directly, using the browser interface offers extensive help text.

Tuning in this section involves editing the HTTP configuration file (such as /www/ap_ow_1/conf/httpd.conf). Start HTTP administrative services by typing on the iSeries server command line: STRTCPSVR SERVER(*HTTP) HTTPSVR(*ADMIN) and hit Enter. Next, in your browser open a session: http://server-name:2001 where "server-name" is the host name of your iSeries server, and sign on. Select IBM HTTP Server for iSeries, and under the Manage tab, select your server. After making your changes, scroll down and click Apply. Then stop and restart your HTTP server instance for these changes to take effect.

6. HTTP threads (ThreadsPerChild): The HTTP Server job runs a pool of threads called ThreadsPerChild for its users. The default, 40, works fine for 400 or fewer users. If you have more than 400 users, change the pool to 10% of the highest expected number of users. For example, set ThreadsPerChild to 80 for 800 users. To verify your thread usage, go into WRKACTJOB and find your HTTP instance's primary job. It is listed under the QHTTPSVR subsystem where the Function column is "PGM-QZSRHTTP". View your threads by hitting F11 twice and take a 5 to display and then a 20 to view the threads. Ensure that there are some extra threads available in the pool (with 0% CPU) during peak usage.

Under the System Resources tab, then Advanced Settings, change the number of threads of threads to process requests. Your HTTP configuration file will now include the line below where the "x" is replaced by the new pool size:

ThreadsPerChild x

Highlights

The HTTP tuning suggestions cover HTTP threads, persistence, caching and logging.

7. Persistence: Without persistence each HTTP request uses a separate TCP connection. Enabling persistence (KeepAlive) allows a single connection to be used for multiple HTTP requests. Reusing a single connection reduces the open and close overhead and improves response time. As shipped, the default Apache server (APACHEDFT) has persistence disabled, but a newly created Apache server sets persistence to enabled. The default settings for persistence worked well for our tests. So, as long as you are not using APACHEDFT as your HTTP server, nothing needs to be changed. To check your settings, go to System Resources and HTTP connections. The Connection Time-out and Max pending settings refer to HTTP server to Java™ server timeouts. The last 3 settings refer to persistence between browser and HTTP server.

8. Caching: Turn dynamic cache on and leave live local cache on. The defaults for "dynamically cache files based on file usage" is off and "update cache when files are modified" is on. We recommend turning on dynamic cache because we saw a slight improvement in response times. Under the Manage tab, use the pull-down to select your server, and then scroll down the left bar and double click on System Resources. Then click on the caching tab and ensure that both boxes under "what to cache" are checked.

DynamicCache On
LiveLocalCache On

9. Logging: Eliminate the unneeded LogFormat lines in your HTTP configuration file and change the log level from "warning" to "error only" logging, for best performance.

a. Double click on the logging tab, then custom formats and remove each log—combined, cookie, agent, referrer, and common. The following 5 lines will be removed from your file (you could also comment them out by editing the file's lines with a #, if you prefer):

Highlights

LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%{User-agent}i" agent
LogFormat "%{Referer}i -> %U" referer
LogFormat "%h %l %u %t \"%r\" %>s %b" common
CustomLog logs/access_log combined

b. At the bottom of the error logs pull-down, change logging level from warning to error, which improves performance considerably. Your configuration file's LogLevel line now looks like this:

LogLevel error

10. HTTP user profile swapping disable: Previously, with the original HTTP server, we had recommended disabling the user profile swapping within the HTTP Server by making a configuration change. Now "disabling the user profile swapping" is the default in HTTP server (powered by Apache), so nothing needs to be done here.

iSeries server

New with V5R1, *TYPE2 directories of the Integrated File Structure offer enhanced performance.

11. Integrated File System (IFS) structure: Change the IFS directories on your iSeries from *Type1 to *Type2.

New with V5R1 of OS/400, *Type2 directories are optimized for performance, size and reliability. With V5R1 and later, there is a Convert Directory API called QP0FCVT2. (New V5R2 systems ship with *Type2 already in place.) You can use this API to check your current settings, have the system estimate how long the conversion will take, and then actually perform the conversion (in restricted state). For more information, go to

publib.boulder.ibm.com/iseries/v5r1/ic2924/info/apis/qp0fcvt2.htm

12. Check for disk activity: On your enterprise server, review the disk subsystem's "% write cache overruns" for a quick check that the number of disk arms is in balance with your overall system.

Highlights

After running Collection Services in iSeries Navigator for 30-60 minutes during peak activity, go to performance tools and run the component report. At the bottom of the disk activity page, review the "% write cache overrun" amount. The overrun should be less than 10-12%, which means that the write cache on the disk I/O card was utilized about 90% of the time. If the overrun is higher, it means that your system may need more disk controllers for improved performance.

Note: Performance Tools (5722-PT1) is required for your iSeries to display this report (or you could write your own SQL statements to analyze the raw data).

13. Additional Tuning Tips: For a more general overview of performance tuning on topics such as memory allocation, system values, other utilization guidelines, prestart jobs, and supplemental tuning tools, see Chapter 9 of the IBM Redbook, *J. D. Edwards OneWorld Xe Implementation on IBM eServer™ iSeries Servers*, SG24-6529 which is available online at http://www.redbooks.ibm.com/

Customizing the ERP 8.0 / OneWorld Xe enterprise server's JAS.INI file improves performance through kernels and security.

ERP 8.0 / OneWorld Xe enterprise server

The JDE™ ERP 8.0 / OneWorld Xe enterprise server provides configurable parameters through the jde member of the ini file in the system library (B7333SYS). On the iSeries server command line, type in: STRSEU system-library-name/INI and hit Enter. Then take a 2 to edit the JDE member. Scroll down to each of the sections that are discussed below and either change the default or add the extra line as indicated. When you are finished, you need to hit F3 to exit and enter to save your changes. Then you need to stop and restart JDE enterprise services (ENDNET, CLRIPC, and STRNET), your JVM, and your HTTP server for the changes to take effect.

Note: Application Development Tools (5722-WDS, Option 21) is required on your iSeries to run SEU.

Highlights

14. Kernel settings: Allow one call object kernel for every 5 to 10 concurrently active users (7.5 is a good place to start), one security kernel job for every 100 users, and one network job for every 20 call object kernel jobs (with a minimum of 2 kernels).

We set the number of call object, network, and security kernel jobs based on the number of users. In our tests, we found that 7.5 users per call object kernel offered us the best performance for the workload that we were supporting. Your ratio might vary if the call object kernels run a different mix of business functions. The rule of thumb for network kernel jobs is to have 1 for every 20 call object kernel jobs. Network kernel jobs should have a minimum of 2 to avoid potential bottlenecks because the first kernel acts as a traffic cop to pass off the requests, as well as performing network requests itself. One security server for every 100 users is recommended.

Replace the “x” below with the number of kernels for your environment. We recommend autostarting your kernels as well.

Network kernels:

[JDENET]
maxNetProcesses=x (default is 1)

Security kernels:

[JDENET_KERNEL_DEF4]
maxNumberOfProcesses=x (default is 1)
numberofAutoStartProcesses=x (default is 0)

Call object kernels:

[JDENET_KERNEL_DEF6]
maxNumberOfProcesses=x (default is 10)
numberofAutoStartProcesses=x (default is 0)

Eliminate extra user profile swaps.

15. ONEWORLD user profile security setting: Eliminate the extra user profile swaps by the call object kernels.

The ERP 8.0 / OneWorld Xe enterprise server runs under the ONEWORLD user profile. When a call object kernel performs a business function, the job swaps to the user profile of the user that is

Highlights

running the business function. If the setting, useAS400Security, is set to TRUE, which is the default, the job then makes additional swaps when accessing iSeries server objects.

To eliminate these extra swaps, you can use a single user profile (ONEWORLD) for accessing iSeries server objects. This change significantly reduces the amount of swapping and improves the overall response time. To implement the change, in the [AS400] section, add the following new line:

```
[AS400]
useAS400Security=FALSE
```

Setting useAS400Security to FALSE causes the JDE iSeries enterprise server to run more like ERP 8.0 / OneWorld UNIX® or Windows (i.e., it runs under a single user profile - ONEWORLD). There is no additional iSeries user security beyond the security imposed by use of the single user profile, ONEWORLD. The security server validates the user id and password passed from the web client.

Monitor call object kernels' health less frequently for faster response times and lower CPU utilization.

16. Check kernel setting: Enable the check kernel health setting in the ini file.

While the enterprise server is running, it passes messages between the network jobs and the kernel jobs. By default, every 10th time it sends or receives a message, the network job verifies that each call object kernel is running. Increasing this setting reduces the amount of overhead within the enterprise server and improves the overall response time. For example, you can choose to check kernels' health every 5,000 or 10,000 times there is a message; the trade-off is better performance versus prompt knowledge of a kernel problem. Monitoring kernel jobs is important but if you have a heavy transaction load, you might consider increasing this number to have it check for kernel health less frequently. Alternatively, you might reduce this number while testing new modifications.

In the [JDENET] section, add the new line, checkKrnlHealth with the

Highlights

Java server tuning covers connection pools, max users, JDENet, timeouts, caching, threads, asynchronous processing, multi-line edits, and JDBC drivers.

value that you choose, as in the example below. Setting the network job to check every 5,000 times it sends or receives a message is a reasonable place to start.

```
[JDENET]
checkKrnlHealth=x
```

ERP 8.0 / OneWorld Xe Java™ server

To optimize performance in the Java™ server edit the following files, by changing the settings listed below. These files, listed with their default locations in the IFS, can be modified through your normal PC editor like Notepad.

\JDEdwards\jas\EA_JDEdwards_1.ear\webclient.war\WEB-INF\jas.ini
\JDEdwards\jas\EA_JDEdwards_1.ear\webclient.war\WEB-INF\HTMLClient.ini
\JDEdwards\jas\EA_JDEdwards_1.ear\webclient.war\htmloverrides.ini
\JDEdwards\JAS\EA_JDEdwards_1.ear\webclient.war\js\Form.js

17. JDBC Connection pool settings in the jas.ini: JDBC is the database interface used between the ERP 8.0 / OneWorld JAS code running on WebSphere and the database enterprise server. These settings affect the number of QSQSRVR or QZDASOINIT jobs that get used. Connection pooling allows web clients to use dramatically fewer of these jobs than fat and TSE clients because they get reused.

If the JDE userids are all mapped to a single iSeries server system user profile (proxy userid), set initial connections to 1 for every 10 users, minimum connections to 5, pool growth to 10, and maximum connections equal to the number of users, up to a maximum of 1500 per instance. Initial connections can be set to one for every 10 users.

```
[CONNECTION POOL]
MaxConnection=x
MinConnection=5
PoolGrowth=10
InitialConnection=x
```

Highlights

If each of the JDE userids is mapped to a separate iSeries server user profile, then set initial connections to 1, minimum connections to 1, pool growth to 1, and maximum connections to 2. (This method for connection pooling may be eliminated for future releases of ERP 8.0.) Example for various counts of users using multiple proxy userids:

```
[CONNECTION POOL]
MaxConnection=2
MinConnection=1
PoolGrowth=1
InitialConnection=1
```

The Max Users setting needs to be set high enough so that no user is denied access to the server.

18. Max Users allowed in the jas.ini file: Set the MAXUser value equal to the maximum number of users to be concurrently connected to the web instance. In the jas.ini file's OWWEB section, replace the "x'" to set MAXUser=maximum users expected. If this value is reached, the next user will be denied.

```
[OWWEB]
MAXUser=x
```

Go into web SAW under the User List pull-down to verify your MaxUser setting and compare it to max users connected during peak activity.

19. JDENET connections in the jas.ini file: Set the maxPoolSize value (default is 50). The connections between the JDE Java™ server and the enterprise server are JDENET connections, the same as any fat client or TSE user. The number of JDENET connections should be the same as the number of users that are being run in the JVM. Go into J.D. Edwards "web" SAW under the JDENet Pool pull-down to verify your maxPoolSize setting. During peak activity, monitor the Waiting column. This number should usually be 0 to ensure that there is no latency for JDENet messaging. To increase your connections, add the following line to the [JDENET] section and replace "x" with your user count.

```
[JDENET]
maxPoolSize=x
```

20. JAS timeout setting in jas.ini needs to exceed the WebSphere

Highlights

timeout setting: The Tech Flash on the Knowledge Garden (GSSTF-02-0392) explains how your UserSession value in the Cache section of the jas.ini needs to be a longer time period than WebSphere's timeout, or your users may experience errors and have to log out and log back in again. Check in WebSphere Administration Console, by going to your JVM, then Services tab, then Session Manager Service, then Advanced, if you need to display your Invalidation timeout setting (default is 30 minutes).

The default for the value in the jas.ini file's UserSession is 1200000 milliseconds (20 minutes). Change it to 2400000 (40 minutes), for example.

[CACHE]
UserSession=x

Caching of Java serialized objects reduces system overhead and improves user response time.

21. Cache Java™ serialized objects in the jas.ini: Add the DDITEM value to turn on caching for Java™ serialized objects and to set the caching time limit. In the [CACHE] section of the jas.ini file add: DDItem=x (where x is the number of milliseconds to hold the cache). For example, DDItem=6000000 preserves the cache for 100 minutes. If you want to guarantee the cache will be held overnight, i.e., 12 hours, the value would be 43200000. Sixteen hours would be 57600000. This setting improves overall performance considerably.

[CACHE]
DDItem=x

Note: Be aware that the first user in each application will experience slow performance because the Java™ serialized objects are being loaded into cache.

Asynchronous processing significantly improves user response time.

22. Virtual Thread Pool Size and asynchronous processing: Go to the [General] section of the HTMLoverrrides.ini file.

a. Change OWVirtualThreadPoolSize to the expected highest number of users (virtual users). During heavy usage, check in J. D. Edwards "web" SAW under Thread List, and JDE_SERVER_LISTENER port, to

Highlights

see how large the pool gets. Set this value higher than the pool size, because if it is ever reached, the next user will be denied.

OWVirtualThreadPoolSize=x

b. Asynchronous business function processing improves throughput because a user can keep working while a previous request is processing in the background. For example, a sales order gets committed in the background while the user is viewing its order number and starting the next order. Although occurring automatically for fat clients and TSE clients, asynchronous processing of business functions must be turned on for web clients. Enable asynchronous business function processing by setting it to TRUE.

AsynchBSFNEnabled=TRUE

c. Change AsyncThreadPoolSize to the number of threads needed to run for asynchronous business functions (like Sales Order Entry). During heavy usage, check in J. D. Edwards "web" SAW under thread list, then under AsynchBSFNThread group. If you are approaching the pool size, then increase it for best performance.

AsyncThreadPoolSize=x

Silent post with multi-line edit improves user response time in applications like sales order entry.

23. Silent post with multi-line edit: While fat clients and TSE users make use of silent post automatically, it must be turned on for web clients via multi-line edit. Silent post happens while a user enters edit lines into forms, each previous line or lines gets processed in the background. So when the user presses OK on the form, the processing of the form is all that remains to be completed. To use silent post function, after enabling asynchronous processing (see previous section), you must turn on multi-line edit.

a. Turn on multi-line edit in each of your applications that would benefit from silent post and regenerate your Java™ serialized objects. The following example turns on multi-line edit in the htmloverrides.ini file for sales order entry and standard voucher entry:

[HTMLAppOverrides]
P4210_W4210A

Highlights

P0411_W0411K

[P4210_W4210A]
multiLineEdit=True
[P0411_W0411K]
multiLineEdit=True

Note: The preferred way to enable multi-line edit is to use the ERP 8.0 / OneWorld design tool Form Design Aid (FDA). There is a document, "SP 17 Multiple Line Edit", on the Knowledge Garden. Search the Knowledge Garden under Services, Advanced Technology for "SP 17 Multiple Line Edit".

b. An option within multi-line edit is to increase the number of rows processed at a time, if your environment warrants it. If you want more than two lines to be processed at a time, change the JavaScript file, Form.js.

The rowCriticalSize variable can be changed to the number of edit lines that should be processed at one time. Although the setting can be as low as 1, the default of 2 might be best in most cases; however, end user response time in our environment improved when the setting was changed to 3. You need to evaluate how the majority of your users operate and test the setting chosen: if many-line forms are the norm, 3 might be better than 2, for example. Since the forms during our testing typically had 10 lines, we set the variable to 3: background processing occurred three times, and then when the user pressed OK, only one more line was left to edit along with the final form processing.

After the comment line, "// asynchronous row processing", change the rowCriticalSize variable to the number that suits your environment:

```
// asynchronous row processing
var rowCriticalSize = x;
```

There are two JDBC drivers available on the iSeries server.

24. JDBC Drivers on iSeries: The iSeries server offers two JDBC drivers for database access. The first is the native driver, which is part

Highlights

of OS/400, and is characterized by QSQSRVR jobs running under subsystem QSYSWRK. With SP21, the native driver is used to access data on the same iSeries server, All-in-One (AIO). See Figure 2.

The second JDBC driver is the Toolbox for Java™ (5722-JC1) driver, which utilizes QZDASOINIT jobs in the QUSRWRK subsystem. The Toolbox driver is used for accessing data on all other systems including another iSeries server--Virtual 3 Tier (V3T). See Figure 3. In the V3T configuration, we used the Mod 5 version of the Toolbox driver: \QIBM\ProdData\HTTP\Public\jt400\lib\jt400.jar

This code, or later versions, can be downloaded from the IBM Java™ Toolbox web site: http://ibm.com/eserver/iseries/toolbox/

To set up JDBC, whether you use the Toolbox driver only or are also adding the native driver, the following 2 lines are used. In the [DB SYSTEM SETTINGS] section, the JDBC driver is set for iSeries (Type=I). The Toolbox driver is then identified in the [JDBC DRIVERS] section of the jas.ini file.

```
[DB SYSTEM SETTINGS]
Type=I
[JDBC DRIVERS]
I/4/R=com.ibm.as400.access.AS400JDBCDriver
```

Using the native JDBC driver and QSQSRVR jobs for AIO is a performance advantage of SP21 over SP20.

The Toolbox driver is used for all configurations of SP20 as well as V3T with SP21. For AIO with SP21, the Toolbox driver alone could also be used; however, we strongly recommend using the native driver. The native driver is faster and more efficient than the Toolbox driver. It is implemented by adding this extra line in the jas.ini file:

```
[DB SYSTEM SETTINGS]
AS400NativeJDBCDriver=TRUE
```

To verify your drivers, start J. D. Edwards “web” SAW, then JDBC Pool, and click on Loaded Drivers. For AIO with SP21, both drivers are loaded because the Toolbox driver loads the native driver and passes all the work to it.

Appendix A – Additional Information

A.1 Disclaimer

Performance data in this document was obtained in a controlled environment with specific performance benchmarks and tools. This information is presented along with general recommendations to assist the reader to have a better understanding of IBM and J. D. Edwards products. Results obtained in other environments may vary significantly and does not predict a specific customer's environment.

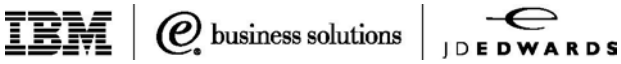
The information contained in this document has not been submitted to any formal IBM or J. D. Edwards test but has been derived based on experience in configuring and tuning a number of different iSeries systems in our test labs. This information is distributed AS IS. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM and J. D. Edwards for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

A.2 Information

If you have questions concerning this document, please direct them to Gerrie Fisk at mcfisk@us.ibm.com or Clark Scholten at scholten@us.ibm.com.

For more information

For additional information on integrated, collaborative enterprise solutions from IBM and J.D. Edwards, call 1 888 426-5505 or visit www.ibm-jdedwards.com/



All performance data contained in this publication was obtained in a specific environment, and is presented as an illustration. The results obtained in other operating environments may vary.

The information in this document concerning non-IBM products was obtained from the suppliers of those products or from their published announcements. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

IBM Corporation
Integrated Marketing Communications
Server Group
Route 100
Somers, NY 10589
U.S.A.
ibm.com

J.D. Edwards World Solutions Company
World Headquarters
One Technology Way
Denver, CO 80237
U.S.A.
jdedwards.com