Deployment Considerations

QAD .NET User Interface

QAD .NET UI 2.5

September 2007

This document contains proprietary information that is protected by copyright. No part of this document may be reproduced, translated, or modified without the prior written consent of QAD Inc. The information contained in this document is subject to change without notice.

QAD Inc. provides this material as is and makes no warranty of any kind, expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. QAD Inc. shall not be liable for errors contained herein or for incidental or consequential damages (including lost profits) in connection with the furnishing, performance, or use of this material whether based on warranty, contract, or other legal theory.

QAD and MFG/PRO are registered trademarks of QAD Inc. The QAD logo is a trademark of QAD Inc.

Designations used by other companies to distinguish their products are often claimed as trademarks. In this document, the product names appear in initial capital or all capital letters. Contact the appropriate companies for more information regarding trademarks and registration.

Copyright ©2007 by QAD Inc.

QAD Inc.
100 Innovation Place
Santa Barbara, California 93108
Phone (805) 684-6614
Fax (805) 684-1890
http://www.qad.com

Contents

CHAPTER 1

Introduction

This document outlines general concepts relating to QAD .NET User Interface installation and tuning options, including specific steps that system administrators can take to create a successful deployment. Although the material here can guide any administrator in making improvements to the QAD 2007.1 software environment, some of the information mentioned is best implemented by experienced personnel.

This document is a good starting point for understanding deployment, stability and performance issues. For complex systems in need of performance optimization or capacity requirements planning, it is strongly recommended that QAD Global Technical Services be contacted for advice.

What is in this Guide?

Use this guide for general information on the various setup and deployment options with the QAD .NET User Interface (UI) 2.5 for QAD 2007.1. This guide is also a useful tool for understanding sizing and hardware selection options.

This document is intended as a supplement to *Installation Guide: QAD User Interfaces*.

Information in this document has been produced as combination of internal QAD testing, and external customer analysis and benchmarking.

Starting with Service Pack 5, QAD Enterprise Applications 2007 (QAD 2007) is the new name of the MFG/PRO eB2.1 product line. MFG/PRO eB2.1 SP5 is QAD 2007 and eB2.1 SP6 is QAD 2007.1.

Audience

These instructions are intended for a system administrator with experience installing QAD software solutions, as well as configuring and managing hardware and operating system software. This person also should have a good understanding of networking concepts and administration, as well as a basic understanding of performance concepts.

If you do not have this expertise within your company, you should contact your QAD Support representative for information on the installation and capacity requirements planning offerings supplied by QAD's Global Technical Services.

Related Documentation

Progress Documentation

Find the complete Progress Open Edge documentation set online at the Progress Web site:

http://www.progress.com/openedge/index.ssp

QAD Documentation

For information on installing QAD 2007.1 or converting to a more recent release, refer to the appropriate installation or conversion guides for your system. Electronic copies of the latest versions are available on the documentation portion of the QAD Web site.

For information on using QAD 2007.1 functions, refer to the User Guides.

For instructions on navigating the QAD .NET UI, refer to *User Guide: QAD .NET User Interface*.

For details about installing the QAD .NET UI Software, see *Installation Guide: QAD User Interface*.

QAD Web Site

The QAD Web site provides a wide variety of information about the company and its products. You can access the Web site at:

http://www.qad.com

For QAD 2007.1 users with a QAD Web account, product documentation is available for viewing or downloading at:

http://support.qad.com

You can register for a QAD Web account by accessing the Web site and clicking the Accounts link at the top of the screen. Your customer ID number is required. Access to certain areas is dependent on the type of agreement you have with QAD.

Most user documentation is available in two formats:

Portable document format (PDF). PDF files can be downloaded from the QAD Web site to your computer. You can view them with the free Adobe Acrobat Reader. A link for downloading this program is also available on the QAD Web site.

HTML. You can view user documentation through your Web browser.

The documents include search tools for easily locating topics of interest.

Features also include an online solution database to help QAD 2007.1 users answer questions about setting up and using the product. Additionally, the QAD Web site has information about training classes and other services that can help you learn about QAD 2007.1.

Symbols and Conventions

This document uses the text or typographic conventions listed in the following table:

If you see:	It means:
`monospaced text`	A command or file name; for example `compile.lst`.
`italicized monospaced text`	A variable name for a value you enter as part of an operating system command; for example, *`QAD UISystemName`*.
`indented command line`	A long command that you enter as one line, although it appears in the text as two lines.
Note	Alerts the reader to exceptions or special conditions.
Important	Alerts the reader to critical information.
Warning	Used in situations where you can overwrite or corrupt data, unless you follow the instructions.

CHAPTER 2

Performance Concepts

This section covers basic performance tuning concepts and system performance management:

Monitoring and Tuning Concepts

Identifying Performance Problems

Knowing When the Hardware is Under Load

Setting Performance Objectives

Monitoring and Tuning Concepts

When users access the QAD Enterprise Applications 2007.1 software, they are placing demand on the computing infrastructure. This demand introduces the possibility that the available resources will come under load and performance can be seen to suffer.

Unless performance is being actively managed and benchmarked, the user expectations of performance will be hard to quantify. Statements such as “the system is running slow” or “my report is taking forever to come out” are not adequate for tracking and reporting on system performance. It is far better to track quantitative information such as “my report is running 65% slower than the established upper boundary”. If the user and technical community do not understand what normal system behavior is, it can be difficult to diagnose what is abnormal.

In a stable, actively monitored, and well documented environment, the software administrators (systems or database) will usually notice performance problems (abnormal system behaviour) as they arise. More often, however, is that performance problems tend to be reported by the user community.

Identifying Performance Problems

When a performance problem is reported, you can begin to identify the performance problem by narrowing down the possible causes:

- A Program Runs Slowly
- The Entire System Runs Slowly at Specific Times
- Everything Runs Slowly at Random Times
- Everything Runs Slowly All the Time
- Certain Interfaces Run Slow
- Remote Access (Wide Area Network) Speed is Unacceptable

A Program Runs Slowly

A program can start to run slowly for a number of reasons. Some simple detective work can usually isolate the primary cause of the problem:

- Has the program always run slowly?
- If the program has just started running slowly, a recent change might be the cause.
    - Has a new version or patch been installed?
    - If so, then check with QAD support or the programmers involved for possible performance impact.
- Has something in the environment changed?
    - Operating sytem patch, additional software, firewall, DNS change, and so on.
- Does the program always run at the same speed or is it sometimes faster?

The Entire System Runs Slowly at Specific Times

There are several reasons why the system may slow down at certain times of the day.

Most people have experienced the rush-hour slowdown that occurs because a large number of people in the organization habitually use the system at certain peak times during the day, week, or month. Examples include early morning distribution functions, mid-week billing functions, or end-of-month accounting functions.

If you are routinely using tools like sar (Linux / UNIX) or Windows Performance Monitor, there is a high chance you have automatically collected performance data for the period that spans the time of the slowdown. You can use this data to analyze the primary causes of system slowdown:

- Are some disks much more heavily used than others?
- Is the CPU idle percentage consistently near zero?
- Is the number of network packets sent or received unusually high?
- Is the system low on available memory?

If the CPU is under load, try and identify the programs being run during this period.

If the slowdown is at an unexpected time (for example, during lunch time), someone might be running an intensive program such as MRP and is trying to run it at the least intrusive time.

You should also check the contents of the crontab or automatic job scheduler for expensive operations scheduled during these periods.

Everything Runs Slowly at Random Times

These situations can be frustrating to track down because of the random nature of the problem. Where possible, instruct users to record the date and times of the slowdown, to both establish a pattern, and to enable you to investigate system performance logs to see what is happening at these times.

Everything Runs Slowly All the Time

A generalized case of slowness typically requires a full system audit and tune up, using the techniques described later in this document. An outcome could be the replacement of one or more hardware resources.

Certain Interfaces Run Slowly

It might be the case that the QAD 2007.1 Character User Interface (CHUI) performs acceptably, but the QAD .NET UI based user interface is considered slow/sluggish.

In terms of raw data entry speeds, The QAD .NET UI might not be as fast as character. The QAD .NET UI, however, provides a much richer and more flexible user experience, and has many productivity enhancements. Even though the comparable speeds within the same screen might show a data entry advantage to the CHUI interface, the overall productivity might be higher with the QAD.NET UI due to the enhanced features and workflow optimizations.

However, if the performance difference between the CHUI and QAD .NET UI is comparatively large, this could an indication that while the underlying operating sytem and database might be tuned quite well to support a multi-user ERP application like QAD 2007.1, there is additional tuning that might be needed at the Java, Tomcat or network levels.

Remote Access (Wide Area Network) Speed is Unacceptable

In addition to using the network tuning techniques in this document, there are some other issues to consider:

- The QAD .NET UI software does not include internal DNS resolution, and might require a WINS server or some other network based domain/IP resolution functionality.
- Could there other network traffic (such as email or content downloads) causing the problem? Many companies have gained performance advantages through use of packet shaping technologies.
- Is the network bandwidth simply not enough to support the required traffic? Is the latency too high? Are the users trying to process through too many network locations?

Knowing When the Hardware is Under Load

A reported performance problem does not automatically mean the hardware resources are under load. The problem might be related to application or database performance, and can be alleviated using a programming or tuning exercise.

To find out when a server is under load, there are some commonly accepted industry standards for defining when a piece of computer hardware is under load. These standards are covered in this section. Note that the following definitions refer to a sustained (or rolling) load rate.

Server load can be monitored using the Performance Monitor on Windows Server, and using `sar –q, top(as), or vmstat` under Linux/UNIX. In addition to the generic UNIX tools, HP/UX provides Glance and GlancePlus for system performance monitoring.

To maintan a decent level of system responsiveness, the %runocc (run queue occupied) should average less than 50%, and the server load should be less than 75% of the total number of processing threads.

For example, on a single CPU, single thread CPU (total of one processing thread), the server load should remain under 0.75 for good responsiveness.

On a dual core, dual CPU server with hyperthreading enabled (total of eight processing threads), the server load should remain under 6.

When a system comes under load, the most common bottlenecks are one (or more) of the following (from the fastest resource to the slowest): CPU, memory, disk drive system, or network. If the bottleneck cannot be resolved by tuning the application or software environment, the hardware bottleneck generally needs to be upgraded (faster CPUs, more RAM, extra disks or disk controllers, faster networking gear, and so on). A principal goal of performance tuning is to move the bottleneck to the fastest resource.

Setting Performance Objectives

When users access the QAD 2007.1 application software, they are placing additional demand on the system (usually requiring both good response time, and high throughput). When the user demand is high, system resources can come under load and might fall out of balance. At this time, performance can be seen to suffer with symptoms such as the following:

- Decreased system throughput (reports and complex processing functions take longer)
- Increased response time (moving from field to field on a screen takes longer)

To track this quantitative performance information, it is required to have an established history of performance of computing (business) functions for comparative purposes. That is, a need to establish benchmarking and monitoring of key QAD processes, and actively record the information over time.

A simple way of determining application performance is to benchmark key applications. A subset of QAD Enterprise Applications programs should be chosen as key performance indicators (KPIs). The KPI applications should be run on a regular basis, at set times, with set selection criteria, with a view to timing how long they take to execute. If data is gathered and tracked for these KPI programs over time, then the following information can be gathered:

- What are normal execution times?
- What are abnormal execution times?
- When do the abnormal execution times occur?
- Is the system, as a whole, getting slower over time?
- At what rate is the system getting slower?
- What shall be the upper limit of KPI program execution time before the system is determined to be suffering performance bottlenecks?

On the purely technical level, system performance needs to be tracked over time, using both spot checking and historical data automatically gathered within set sampling periods. Using this information system performance trends in both the medium term (for instance, “what do we suspect the system will behave like two months from now”) and the short term (for instance, “why is the system currently performing badly?” or “why does the system perform badly from 3pm to 5pm daily?”) can be established.

Identification of Critical Resources

In general, the performance of a given workload is determined by the availability and speed of one or two critical system resources. The analyst must identify those resources correctly or risk falling into an endless trial-and-error operation.

Systems have physical, logical, and possibly virtual resources. Physical resources are generally easy to identify, with many system performance tools available to assess their utilization. The real resources that most often affect performance are as follows:

- CPU

- Memory
- Disk I/O
- Various adapters
- Disk space
- Network access

Logical resources (such as databases, application servers, and web servers) also need to be identified. Logical resources are generally application level abstractions that provide the business functions to the users. A set of physical disks set up in a RAID configuration may also presented to the operating system as a logical device.

A virtual resource relies on emulation to simulate underlying hardware, or an operating environment.

It is important to be aware of logical and virtual resources as well as physical resources. Processing threads can be blocked by a lack of logical resources as well as by a lack of physical resources, and expanding the underlying physical resource does not necessarily ensure that additional logical resources will be created.

Continuous Monitoring

There are several advantages to continuously monitoring system performance:

- Detect underlying problems before they have an adverse effect
- Allows you to establish a baseline for comparison
- Maximize efficient use of computing resources
- Minimize surprises
- Predict when a hardware upgrade may be needed

The alternative to continous monitoring is ad-hoc or minimal monitoring. The key attributes of a minimal monitoring model are as follows:

- It is usually cheaper
- Performance issues often arise without warning, leading to emergency situations
- Lack of transparency at the management level

CHAPTER 3

System Configuration and Tuning

This section covers basic performance tuning concepts and system performance management. It also provides information on the various configuration and setup options available with the QAD .NET User Interface.

Disclaimer

Hyperthreading and Simultaneous Multithreading

Redhat and Novell (SuSE) Enterprise Linux

HP/UX

IBM AIX

Sun Solaris

Windows Server

Network

Tomcat

Citrix Presentation Server

Database and Progress Software

QAD .NET User Interface Options

Disclaimer

The information presented in this section is informational only, and is presented as a guide. Every system is unique and may have unique configuration and setup requirements.

Hyperthreading and Simultaneous Multithreading

Hyperthreading support is enabled at the BIOS level on PC class Linux (and Windows) servers. The following observations have been made on Hyperthreading and enterprise Linux:

- Hyperthreading introduces overhead to maintain the virtual process threads.
- On a light to medium loaded system, hyperthreading enabled systems may be slower than non-enabled systems.
- On a CPU bound system, hyperthreading may provide some performance gains of up to 25% for a two processor system.
- The above performance gains drop away as additional processors are added.
- Hyperthreading is not recommended for eight processor or above systems (8 by single core, 4 by dual core, 2 by quad core).

Simultaneous Multithreading (SMT) is a new feature available on POWER5/6 systems running AIX 5L V5.3+ or IBM i5/OS® Version 5, Release 3+. SMT allows two instruction paths to share access to the POWER5 execution units on every clock cycle, similar to the way that hyperthreading works on a PC based server. Similar performance rules apply to SMT as they do with PC based hyperthreading.

Redhat and Novell (SuSE) Enterprise Linux

UNIX and Linux installations must typically be configured to support multi-user database systems. When installing from media, the operating system resources available are usually not sufficient to support applications such as those that QAD provides. Most of the actions in this section require superuser (root) access.

File Handles

The total amount of allowable processes and open files may need to be raised. This is controlled in the file `/etc/security/limits.conf`. An example of the increased limits is shown below.

```
#<domain>      <type>   <item>   <value>
#
*              soft     nproc    2047
*              hard     nproc    16384
*              soft     nofile   1024
*              hard     nofile   65536
```

Shared Memory and Other sysctl Tunables

The default kernel settings for Linux are generally tailored for desktop workstations, and are under resourced for a relational database system. The kernel settings are stored in the `/proc` file system, and some settings can be modified at runtime by the `sysctl` function.

`/etc/sysctl.conf`

The following changes made to `/etc/systctl.conf` will support at least a 125 user system on Redhat and SuSE Linux:

```
kernel.core_uses_pid = 1
kernel.shmmax = 2147483648
kernel.shmmni = 4096
kernel.shmall = 2097152
kernel.sem = 500 32000 200 256
fs.file-max = 65536
net.ipv4.ip_local_port_range = 1024 65000
```

The kernel can be updated in real time to make these settings take effect by executing the following command:

```
# sysctl –e –p /etc/sysctl.conf
```

Bigpages

Note that internal testing has shown that enabling *bigpages* (large virtual memory page sizes) support has shown a reduction in application performance, and its use is not recommended.

The following kernel settings should be set to 0 to ensure bigpage support is switched off:

Novell: `kernel.shm-use-bigpages = 0`

Redhat: `vm.hugetlb_pool = 0`

Kernel Deadline Scheduler

Starting with the 2.6 kernel, the I/O scheduler can be changed at boot time to control the way the kernel commits reads and writes to disks.

The Completely Fair Queuing (CFQ) scheduler is the default algorithm in enterprise Linux's which is suitable for a wide variety of applications and provides a good compromise between throughput and latency. In comparison to the CFQ algorithm, the Deadline scheduler caps maximum latency per request and maintains a good disk throughput which is best for disk-intensive database applications.

Hence, the Deadline scheduler is recommended for database systems.

To switch to the Deadline scheduler, the boot parameter `elevator=deadline` must be passed to the kernel that's being used. Edit the `/etc/grub.conf` file and add the following parameter to the kernel that's being used, for example:

```
kernel /vmlinuz-2.6.18-8.el5 ro root=LABEL=/ rhgb quiet
crashkernel=128M elevator=deadline
```

File Handle Limits

The total number of allowable processes and open files may need to be raised. This is controlled in the `/etc/security/limits.conf` file.

An example of typical raised limits is as follows:

```
#<domain> <type> <item> <value>

* soft nproc 2048
* hard nproc 16384
* soft nofile 1024
* hard nofile 65536
```

HP/UX

Tuning the HP/UX Kernel

With HP/UX, kernel parameters are most easily changed using the Systems Administration Manager (SAM) tool:

1. # sam
2. Choose Kernel Configuration > Configurable Parameters
3. Highlight the parameter to change
4. Choose Actions > Modify Configurable Parameter
5. Enter the new value
6. Choose OK

Set the Dynamic Buffer Cache (Kernel Setting)

`dbc_min_pct` defines the minimum percentage of memory to be used by dynamic buffer cache.

`dbc_max_pct` defines the maximum percentage of memory to be used by dynamic buffer cache.

If `dbc_min_pct` is set to too low a value, very high demand on the buffer cache can effectively hang the system. This is also true when using fixed buffer cache. `dbc_max_pct` must be greater than or equal to `db_min_pct.`

By default, these values are set too high (5/50) for a relational database system. They should be lowered to more reasonable values.

A good starting point would be the following:

```
dbc_min_pct = 3

dbc_min_pct = 10
```

Other Kernel Tunables

HP/UX default kernel settings are usually set too low for relational database systems, particularly in terms of shared memory and the number of files that can be opened. The following are a good set of initial values to use. As always, research and test the effects of any kernel level change before applying to production systems.

Create_fastlinks	1
Maxfiles_lim	8196
Maxfiles	4096
Maxusers	512
Max_async_ports	256
Max_thread_proc	16384
Nproc	4116
Maxuprc	512
Msgmap	2048
Msgssz	512
Msgseg	32767
Msgmnb	65535
Msgmax	131070
Msgmni	50
Nfile	58145
Nflocks	8000
Ninode	60000
Npty	2024
Sema	1
Semaem	16384
Semmns	16384
Semmsl	4096
Semmni	2048
Semmap	2050
Semmnu	1536
Semvmx	32767
Shmem	1
Shmmax	2147483647
Shmmni	1024
shmseg	1024

IBM AIX

The following has a critical performance impact on the QAD .NET UI and it is strongly recommended it be implemented.

AIX systems have been shown to include large delays in sending information to the QAD Desktop and QAD .NET UI. This is due to the 200ms default delayed acknowledgement setting at the TCP layer.

To fix this problem:

1. Login and su to root
2. # mit (to enter systems management)
3. Choose performance and resource scheduling
4. Choose Tuning Kernel and Network Parameters
5. Choose Network Option Parameters
6. Change the fasttimo parameter from 200ms to 50ms.
7. Change the tcp_nodelayack parameter to 1 (that is, switch it on)

Sun Solaris

Tuning /etc/system

These settings are primarily for the network adapter and the high-water shared memory and semaphore settings required for the database with a high number of connections. Consider making these changes to `/etc/system` and reboot:

```
# Recommended Initial Network settings and shared memory

        set sq_max_size=100
        set ip:ip_squeue_bind=0
        set ip:ip_squeue_fanout=1
        set ipge:ipge_tx_syncq=0
        set ipge:ipge_taskq_disable=0
        set ipge:ipge_bcopy_thresh=512
        set ipge:ipge_dvma_thresh=1

        set msgsys:msginfo_msgmni = 128
        set semsys:seminfo_semmni = 4096
        set shmsys:shminfo_shmmax = 2147483648
        set shmsys:shminfo_shmmni = 4096

# Increasing file handles for higher user counts
#
        set rlim_fd_max=16384
        set rlim_fd_cur=8192
```

Windows Server

Reduce Services

There are some common sense items that should never be run on a database server:

- *DNS server* – if for any reason the server needs to be stopped for maintenance, domain naming will no longer work.
- *Domain Controller* – for the same reasons as the DNS, except with greaternegative enterprise consequences. The database server should not act as a primary or secondary domain controller.
- *Router* – the database server should not act as a network router.
- *File and/or print server* – the database server should not act as a file and/or printserver since these functions can consume CPU and memory resources as well as network bandwidth.
- *Terminal Services/Citrix Services* – these types of services require a large amount of memory, depending on user load.

Disable the following services. These services are unnecessary for a QAD 2007.1 server and utilize valuable system resources:

- *License Logging Service*
- *Plug and Play*
- *Remote Access Autodial Manager*
- *Remote Access Connection Manager*
- *Remote Access Server*
- *Telephony Service*
- *Unused Network Protocols*. You should remove all unused network protocols, thus only keeping the protocol that you are using with the database (TCP/IP).

Increase Available Memory for I/O Locking

By default, Windows limits the amount of memory that can be set aside for I/O locking operations to 512 KB. The *maximum* kernel setting for the lock limit is (physical memory minus 64 MB).

Key: HKLM\SYSTEM \CurrentControlSet \Control \Session Manager\Memory Management
Value: IoPageLockLimit
Data type: REG_DWORD
Range: 0 (default) to 0xFFFFFFFF (in bytes — do *not* exceed this maximum!)
Recommendation: (depends on RAM — see Table 11-7 above)
Default: 0x80000 (512 KB)
Value exists by default: No, needs to be added.

Recommended settings:

Amount of physical RAM	IoPageLockLimit Setting (hex)
1 GB (1024 MB)	0x3C000000
2 GB (2048 MB)	0x80000000
4 GB (4096 MB)	0xFC000000
8 GB (8096 MB) and above	0xFFFFFFFF

For more information:

http://www.microsoft.com/technet/prodtechnol/windows2000serv/reskit/regentry/29932.mspx?mfr=true

Apply Latest Reliable Windows Server Service Pack

Ensure all the latest service packs and security patches are current at all times.

Network

Packet Shaping

To maximize the efficiency of the QAD 2007.1 software across the corporate network, it is necessary to use packet shaping software to give the QAD 2007.1 TCP packets priority of general office based network packets (such as email and HTTP packets).

Domain Name System (DNS)

QAD 2007.1 delegates responsibility for translating hostnames into IP addresses to network based DNS resolvers. It might also be necessary for a Windows Internet Name Service (WINS) to be available if pre-Windows 2000 clients or servers need to resolve hostnames.

Tomcat

Tomcat is the Java application server component of QAD .NET User Interface. Typically the memory allocation of Tomcat must be configured to support the Interface requirements.

In the Tomcat installation directory, the following file can be modified to change the memory settings:

[tomcat install directory]/bin/setenv.sh

The following changes are useful for all user counts. Note that the first two settings will increase as the user count increases (see guidelines below).

```
-Xms256m  (minimum heap size)
-Xmx256m  (maximum heap size)
-Djava.net.preferIPv4Stack=true
-Xss128k  (thread stack size)
-XX:MaxPermSize=128m
-XX:+UseParallelGC
```

On non-AIX systems, it is also recommended to use:

```
-server
```

These are the recommended settings for increasing user counts:

Up to 30 users:

```
-Xms256m
-Xmx256m
```

Up to 80 users:

```
-Xms512m
-Xmx512m
```

Up to 120 users:

```
-Xms1024m
-Xmx1024m
```

Over 200 users

```
Xms2048m
Xmx2048m
```

Try and keep `-Xms` equal to `-Xmx`. Having the initial size of the heap be the same as the maximum size saves time needed to expand the heap as an application grows. It can also avoid delays in acquiring the needed memory from the operating system.

Increasing the heap size may decrease the total amount of time spent garbage collecting, but larger heaps typically entail longer garbage collection checkpoints. The proper configuration is determined by the longest acceptable pause time and the amount of memory available for the JVM. You need to tune the sizes of the heap spaces such that you maximize throughput for the machine on which the application will run.

Calculating Total Memory Requirements

Tomcat memory requirements can be calculated by the following formula:
(heap size) + (number of threads)x(thread stack size) = (total RAM used by JVM process).

3.8 GB HP/UX HotSpot Maximum Heap Size Barrier

For large QAD .NET UI installations (over a thousand users), it might be necessary to overcome the 3.8 GB Heap Size limitation when setting the Java heap size. To achieve this, use the `-d64` setting in `setenv.sh`.

Maximum Threads

The maximum number of allowable Tomcat threads with QAD 2007.1 is 150 by default. This value is set in the following file in the `<connector>` element:

`[tomcat install directory]``/bin/server.xml`

Change `maxThreads="150"` to the desired value to support an increased number of processing threads.

Multiple Tomcat Instances for Performance

Customer experience has shown that with concurrent user counts over 250 it may be necessary to run multiple instances of Tomcat to serve the same QAD 2007 application. To achieve this solution, the easiest way is to:

- Have multiple clean installations of Tomcat in separate directory structures
- Each Tomcat installation runs off a different port number
- Each Tomcat installation contains the same QAD 2007.1 web application and database back end
- Clients connect to different Tomcat port numbers through either a manual or automatic load balancing mechanism

Citrix Presentation Server

The QAD .NET UI fully supports the Citrix Presentation Server (previously known as Citrix Metaframe) solution for delivering applications to thin clients. There are various reasons to consider a technology such as Citrix, but this section focuses on some of the performance aspects.

Citrix can be of use in highly latent (> 250ms) or widely dispersed environments to help achieve acceptable end user performance.

Citrix Presentation Server Version

For performance reasons, at least version 4 of the presentation server should be used. The following enhancements were added to Citrix Presentation Server version 4.0:

- Support for Windows Server 2003 x64
- CPU Utilization (throttling power users) management
- Extensive Virtual Memory optimizations (including DLL reuse and reduced paging)

In addition, version 4.5 of the presentation server contains some considerable performance and scalability enhancements when using x64 based Windows 2003 Citrix Servers by eliminating the 4GB maximum memory usage constraints.

Maximizing ICA and RDP Performance

To maximize ICA and RDP performance:

- Disable wallpapers, menu animations and desktop themes
- Disable any RDP or ICA virtual channels not being used (local drive access, printers)
- Use the lowest acceptable resolution and color depth

Speedscreen

If running Presentation Server 4.0 or above, then enable the SpeedScreen technology. This technology uses different kinds of caching and character generation methods to make user sessions appear faster in highly-latent environments.

Database and Progress Software

There are no special requirements for tuning the QAD databases when running the QAD .NET UI.

The normal database tuning techniques should be applied. Consult the QAD installation guides for more information. QAD Global Technical Services can provide comprehensive and detailed database tuning.

A Note on the Appserver Agents

The only accurate method to determine the correct ratio of Appserver agents versus number of QAD .NET UI users is to monitor the particular customer environment.

All customers are different in the way their organization works, and the way their users behave. What works for a company that manufactures heavy machinery would not work for a company that makes and distributes fast moving bakery foodstuffs.

One way of determining the optimum setup for Appserver agents is to use a ratio of 1 to 20 (5 agents for 100 concurrent users) and monitor how the Appserver behaves. The Appserver will start extra agents up to the maximum when it needs to, and will also trim agents when possible.

If more agents are being used than the defined initial amount, then add more initial agents in the `ubroker.properties` file until a good working balance is achieved.

QAD .NET User Interface Options

Some of the performance related QAD .NET UI configuration options are discussed in this section. For further information, please contact QAD support or Global Technical Services.

System Design and Technical Components

For information on how the QAD .NET UI is designed and how the various technical components relate to each other, please consult *Installation Guide: QAD User Interfaces*.

The .NET runtime supports the specification of name-value pairs in an XML formatted document that is associated with application executable via a file naming convention. This basic structure can be extended to support customized data structures as long as they can be represented as well-formed XML. Within a .NET application, the basic and custom configuration information may be accessed via standard API's provided by the runtime.

A configuration consists of configuration parameters. A configuration parameter may be loosely defined as a named value that is passed into the client to alter the behavior of the client without requiring recompilation of the client. That general definition will be further restricted in the remainder of this document to apply only to configuration information that is defined through one of the following sources (in order of precedence):

1. Command line
2. QAD .NET UI client configuration file
3. URI

The value assigned to a parameter via the command line will take precedence over a value assigned to the parameter in the QAD .NET client configuration file.

Additionally, the client allows users to change the value of certain configuration parameters through a user interface accessed from the menu Tools::Options. Please contact QAD support for futher information on how these user preferences can be configured.

QAD .NET UI Client Configuration File

Configuration parameters can be defined in the standard QAD .NET UI client configuration file for the client executable (`$InstRoot\container\QAD.Client.exe.config`) by adding keys to the appSettings element.

URI

This server side configuration document can be edited to make the URI level configuration changes take effect.

[homeserver-root]/configurations/*[system name]*/client-session.xml

Parameter Usage

A canonical representation of configuration names serves as the foundation for handling configuration parameters uniformly regardless of their source. The rules governing this representation are defined below:

Any of the characters [/-.] may be used to separate a name into multiple parts. A name composed of multiple parts is called a compound name.

The characters [-.] in a compound name will be replaced with the character [/].

The characters in a name will be converted to lowercase.

Each of the following representations is equivalent in specifying the value `DEBUG` for the configuration parameter with the canonical name log/level:

Command Line:

```
QAD.Applications.exe –log-level:DEBUG
```

QAD .NET Client Configuration File:

```
<?xml version="1.0" encoding="utf-8" ?>
<appSettings>
<add key="Log.Level" value="DEBUG" />
</appSettings>
```

XML Document:

```
<?xml version="1.0" encoding="utf-8" ?>
<Config>
    <Log>
       <Level>DEBUG</Level>
    </Log>
</Config>
```

Arrays

You can define within a single configuration source a collection of configuration parameters with the same name. In the following example, the “disable” parameter is repeated to specify a set of plugins that should be disabled.

Command Line:

```
QAD.Applications.exe –disable:qad.plugins.plugina -
disable:qad.plugins.pluginb -perf
```

QAD .NET UI Client Configuration File:

```
<?xml version="1.0" encoding="utf-8" ?>

<appSettings>
<add key="Disable" value="qad.plugins.plugina />
<add key="Disable" value="qad.plugins.pluginb />
</appSettings>
```

Maximum Count on Browses

The QAD .NET UI provides functionality that shows the user the total number of records available in a specific browse. This is a useful feature; however, for large tables and record counts, it can be very computationally expensive. Because even 100,000 records is an extremely large number to work with, QAD has set 50,000 as the default maximum records to be counted. If the record count in the browse is above the maximum record threshold, the information displayed in the top right corner of the browse (the record count area) is amended.

This value can be made higher or lower, depending on both the user requirements and the effect the counts are having on server load. The value is set on the server, and can be overridden on individual clients. Be aware that raising the setting to (much) higher values can have an adverse effect on overall system performance.

Server Side

The value to change is:

```
<add key=”MaximumBrowseRecordsToCount” value=”10000” />
```

Maximum Record Download/Print/Export Limit

In addition to the record count maximum, there are also defaulted maximums for the number of records to download, the number of records to print, and the number of records to export to Microsoft Excel.

These values are:

```
<add key=”MaximumBrowseRecordsToDownload” value=”50000” />
<add key=”BrowseRecordsForPrintWarning” value=”10000” />
<add key=”BrowseRecordsForExcelWarning” value=”10000” />
<add key=”ChartElementsForChartWarning” value=”50” />
```

Setting the first two values too high can cause the client computer to freeze up as it attempts to process enormous amounts of data. Because of the record limitation of Microsoft Excel, the Export to Microsoft Excel option should never be set above 65,535 records. The “maximum records to download” is a hard limit and will cause the operation to abort.

Rendering charts with many elements can be CPU intensive. To prevent users from attempting to render large charts that could degrade their computer’s performance, the system checks the number of elements in a chart before rendering the chart and provides a warning if the number of elements might degrade performance. In a chart without grouping, the number of elements in a chart is the number of records in the browse display. In a chart with grouping, the number ofelements is the number of groups of records. The default value for the maximum number of elements allowed before a warning is displayed is 100. This value can be overridden with the `<ChartElementsForChartWarning>` element.

Private Read-Only Buffers

The buffer pool reduces disk I/O by enabling the database to access recently used records from memory instead of reading from disk. Once the buffer pool becomes full, the least recently used (LRU) buffers are flushed to disk. This works quite well for random database access.

However, heavy sequential database access can monopolize the database buffer pool. These types of operations can, therefore, reduce the overall efficiency of the database by forcing random access database operations to read from disk.

Private read-only buffers do not participate in the LRU scheme. Applications that read many records quickly can benefit from private read-only buffers. The new QAD .NET UI browses could (depending on overall application usage) benefit noticeably from the use of private read-only buffers.

These read-only buffers do receive an allocation from the overall (`-B` startup parameter) public buffer pool, up to 25% of the total pool. Private read-only buffers are controlled by the `–Bpmax` startup parameter.

Any browse available in the QAD .NET UI can be made to use private buffers by inserting the program name into the `browse-buffer-config.dat` file:

`[Desktop install Directory]/com/qad/shell/interface/browse-bufferconfig.dat`

```
/* This file is used for configuring the number of private buffers to use    */
/* for browses running in the AppShell. This is used for performance tuning. */
/* This number will be assigned to _MyConnection._MyConn-NumSeqBuffers.      */
/* In general, a good number to use is 6 ; consult Progress documentation    */
/* for more detailed calculations to use for assigning this value.           */
/* The format of lines in this file is: <browse program name> <num buffers>  */
/* For example: pp013 6                                                      */
pp013 6
so009 6
```

Logging Settings

There is a threshold limiting the type of information that will be logged for all logging contexts in which a threshold has not been set.

Note: This value specifies a default for all logging contexts and is set to `Info` if not specified. This value can be set to one of the following values: `Off`, `Error`, `Warn`, `Info`, and `Debug`. Thresholds can be applied to specific logging context using the standard .NET mechanism for controlling diagnostic switches.

For example including the following snippet in the QAD .NET client configuration file would set the context to only log errors:

```
<appSettings>
 <add key="Log.Level" value="1" />
</appSettings>
```

When specifying values in the QAD .NET client configuration file values must be supplied as integers where the following mapping should be used:

```
Error = 1
Warning = 2
Info = 3
Debug = 4
```

An example for changing the logging level in the URI is as follows:

```
<Config>
    <Log>
         <Level>ERROR</Level>
    </Log>
</Config>
```

There are further options for controlling the log files:

`log/file`	The location of the file where logging output should be written.
`log/file/size`	The maximum size of the log file in kilobytes. Note: The default value is 5000 KB. When the maximum size is reached. The file is renamed as a backup and a new log file is created.
`log/file/backups`	The number of log file backups that should be retained. The default value is 5. When the maximum number of backups is reached, the oldest log file backup is deleted.

Timing Metrics

Performance metrics can be collected and printed to the log file. This feature supports an optional integer value that specifies a threshold in milliseconds to limit the output to timed tasks whose duration was equal to or greater than the threshold.

Example for setting this metric using the command line:

```
-perf
-perf:1000   (only shows events that took longer than a second)
```

Example using the QAD .NET UI client configuration file:

```
<appSettings>
   <add key="Perf" value="1000" />
</appSettings>
```

These metrics do not provide the coverage of a code profiler, but some key timespans are captured such as dispatched events. This information can be very useful in debugging performance issues.

Multiple Clients

In certain situations it might be required to limit the number of instances of the client running on a single client Desktop.

`allow/multiple`	When `false` a check will be made to see if the client is already running on the target computer. If it is, the user will be notified, and the existing instance will be activated. The default value is `true`.

Ping (Network Latency Test) Utility

There exists a timing utility in the QAD .NET UI accessed using `http://[host:port]/WebApp/util/ping.html` that shows the round trip time of a data packet going uisng the Tomcat webapp (that is, it gives delivers an approximation of the network latency the user would experience when running the QAD .NET UI screen from Tomcat).

Heartbeat URLs for Load Balancing

System administrators setting up a load balancer for multiple Tomcat instances can access heartbeat URLs to checkconnection status for each instance.

Load balancing scripts (or other custom status scripts) can access the following links under `http://`*`TomcatHost`*`:`*`TomcatPort`*`/heartbeat/`:

status.jsp — returns a status message number of agents that are All, Idle, Init, or Busy.

idle.jsp — returns a page containing the number of idle connections.

busy.jsp — returns a page containing the number of busy connections.

init.jsp — returns a page containing the number of initializing connections.

The pages are installed in *`TomcatInstallDir`*`/webapps/heartbeat/.`

CHAPTER 4

Sizing / Capacity and Deployment Models

This section provides general guidelines that should be considered when selecting hardware and deployment options for the QAD .NET User Interface.

System Sizing

Single/Two/Multi-Tier Installation

Network Considerations

Client Sizing

System Sizing

Sizing Inputs

To adequately size a QAD .NET UI system, a range of factors need to be considered. These are known as sizing inputs. It may difficult to define the inputs for sizing. For example, the range of sizing requests may be as simple as “what do I need to run a .NET UI system with 20 users”, to a formal exercise with with hundreds of pages of statistics about the required sizing.

The list of sizing inputs includes the following:

- Database technology (Progress or Oracle)
- Raw data or disk size in gigabytes (GB) or numbers of records and record sizes
- User numbers and user types (casual users / power users / executive users)
- Network details such as existing topology or bandwidth requirements
- Transaction type and size plus the details of rates (per second, minute, hour)
- Growth in terms of company expectations, data size, or number of users
- Disk technology (direct attached, SAN, NAS) , RAID, and disk type preferences
- Preferences for large monolithic systems (scale up) or clusters of smaller systems (scale out)
- Tiering preferences
- Preferences for a certain hardware vendor or operating system type
- Security constraints
- Requirements based on the customer’s Service Level Agreements (SLAs) with their user population. Redundancy, High Availability, and Disaster Recovery may all affect the final outcome.

System sizing is an iterative process.

Capacity Requirements Planning

Capacity planning is a method of determining the best server and equipment to meet current and future workload and performance requirements at appropriate cost. Capacity planning is only relevant to existing QAD 2007.1 environments, and cannot be used for first-time installations at a site that has never used QAD before.

The capacity of the QAD 2007.1 system is determined by the workload placed upon it. Once the relationship between capacity and current workload has been determined, the same relationship can be used to plan what capacity is required for a higher workload.

The method to plan the capacity of the server is as follows:

- Measure the current utilization of the subsystem (Memory, CPU, Disk) under study
- Decide if the current capacity is enough. Is there currently a bottleneck in the subsystem being studied?
- Characterize the workload
- An understanding of the workload (the QAD 2007.1 application) that must be supported is required.
- Build the capacity-planning model

This is where we establish the relationship between the workload and the capacity required. There are many different models that can be used, all of various levels of mathematical complexity. Simple linear regression techniques (that is, graphs) are often satisfactory, however.

The model can be used to predict the required capacity (hardware) for future workloads, or what is required to handle the existing workload.

Hardware Considerations (CPU)

Some of the below CPU configurations have not been tested but are based on a calculation from CPU types that were tested in the Product Lab.

Because every QAD 2007.1 installation will likely be unique, it is not possible to provide absolute specifications for CPU processing power required without conducting a formal capacity requirements plan.

However, internal testing using sample user workloads and user counts have enabled QAD to provide the following broad guidelines. Initial testing has been done on a Linux server environment. Approximate equivalent CPUs are provided for commercial UNIX hardware.

Very Light User Loads (1 to 10 concurrent users)

Linux: 1 x Dual Core Intel Pentium D 3GHz or AMD equivalent

Windows: 1 x Dual Core Intel Xeon 2.13 GHz or AMD equivalent

Light to Medium User Loads (10 – 150 concurrent users)

As the user count increases, so does the capacity requirements of the CPUs. For more than 10 users, a two-way system is really needed to maintain good response time. On a multi-user database system, it is usually better to have multiple CPUs instead of a single very fast CPU – because a single user can potentially tie up the processor if they run a very intensive process. A dual Xeon based Linux server should be able to handle up to about 75 users with good performance. This is heavily dependant on the usage profile of the application. Some 75-user systems will run very well on a dual Xeon server, where others may struggle.

Linux:	2 x Intel Xeon Dual Core 2.4 GHz or AMD equivalent
Windows:	2 x Intel Xeon Quad Core 1.6 GHz or AMD equivalent
AIX:	4 Core POWER 5/6 processor (p series) or equivalent system I LPAR based solution.
HP/UX:	2 x PA-8900 1 GHz or 2 x dual core 1.4 GHz Itanium 2
Sun/OS:	2 x 1.5 GHz UltraSPARC IIIi processors

Medium to Heavy User Loads (150+ Concurrent users)

For heavier user counts, a formal capacity requirements planning or systems sizing approach should be taken. QAD Global Technical Services can help with this process if required.

Server CPU Requirements (Existing installations)

For customers running previous versions of the QAD .NET UI, it is not considered that upgrading to version 2.5 will require any additional CPU requirements, if all other sizing inputs remain the same.
For customers running the Desktop 2 interface, the estimates on additional CPU requirements is that an extra 25% more processing power is needed.

*Expected CPU Requirements = Current CPU Requirements * 1.25*

Example 1:

- Customer is running a 30-user system on a dual Intel 3.4 GHz Linux server with hyperthreading disabled.
- Average system load is 2. (The average number of jobs server run queue. A measure of the number of tasks the CPUs are trying to manage at the same time)
- Average CPU usage is 40%, with peaks of 75%.
- Expected new system load is (2 x 1.25) = 2.5 (target maximum would be 3)
- Expected new CPU usage is (40% x 1.25) 50% with peaks of (75 x 1.25) of 94%.

- If all other variables remained equal (user count, number of databases), then it is expected the customer could remain on the same server.

Example 2:

- Customer is running a 30-user system on a dual Intel 3.4 GHz Linux server with hyperthreading disabled.
- Average system load is 2.4.
- Average CPU usage is 70%, with peaks of 100%.
- Expected new system load is (0.5 x 1.25) = 3 (target maximum of 3)
- Expected new CPU usage is (70% x 1.25) 87.5% with peaks of (100 x 1.25) of 125%.

In this instance, the CPU usage is quite high already, and it would be expected that the installation QAD 2008 would require the upgrade of the CPU capacity of this server.

Hardware Considerations (Disk Subsystem)

The QAD .NET UI does not generate much disk activity. Disk I/O constraints will generally be caused by other QAD 2007.1 activities such as maintaining the databases.

Disk Space

QAD .NET UI Interface (MFG/PRO extensions):	~ 45 MB per interface
Tomcat:	~ 10 MB
Tomcat QAD .NET UI Web application:	~ 750 MB per interface
QAD 2007	~ 2GB
Databases	as currently used

Hardware Considerations (Memory)

An operating system process consists of two parts – a private data segment unique to each process, and a shared library data segment that is used by multiple processes. For the purposes of providing memory requirements guidelines, a very conservative approach is taken here. It is better to over specify the capacity than under specify.

The Progress executables, for instance, can vary from as low as 15 MB to as high as 100 MB depending on the nature of the work they are doing. The actual size of the process in memory (the memory size of the process minus the memory size of the shared libraries) can vary upwards from 5 MB. A memory footprint of 25 MB is chosen for the purposes of this memory sizing exercise. On specific customer systems, these assumptions should be carefully measured and evaluated to see if they are still applicable.

Below are the calculated memory requirements for components of the QAD .NET UI. These numbers can be used to derive the memory requirements for a specific installation.

Progress executable	~ 25 MB per process
AppServer	~ 45 MB per AppServer
Databases	As per normal Progress database tuning
Java	Maximum heap size for Java config in Appserver
Tomcat	From 256 MB (default) up to 1024 MB and higher
Telnet daemons	0.5 MB each
Login daemons	0.5 MB each
Bash shells	1 MB each
Operating system	Varies, usually about 100 MB

At a simplistic level, basic multiplication can be used to determine expected maximum memory requirements (example only):

	20 users	100 users
Progress executable	20 x 25 MB = 500 MB	100 x 25MB = 2500MB
AppServer	2 x 45 MB = 90 MB	5 x 45MB = 225 MB
Databases	160 MB	160 MB
Java	2 x 50 MB = 100 MB	5 X 50MB = 250 MB
Tomcat	256 MB	1024 MB
Telnet daemons	0.5 x 20 MB = 10 MB	100 x 0.5 = 50 MB
Login daemons	0.5 x 20 MB = 10 MB	100 x 0.5 = 50 MB
Bash shells	1 x 20 MB = 20 MB	100 x 1 = 100 MB
Operating system	100 MB	150 MB
Total:	approximately 1.25 GB	Approximately 4.5 GB

Hardware Considerations (Network)

Under peak conditions, and assuming no other network traffic, the following user counts could reasonably be supported over a local area network (LAN) connection:

- 50 – 500 clients on 100 Mbit Switched.
- 500 clients on a mix of 100 Mbit Switched and Gigabit Ethernet. The server itself should be on a Gigabit Ethernet switch.

Latency

Reduced throughput and increased network latency slow down the response times of the QAD .NET UI application. As a general rule, a latency of 200ms slows down response times (compared to a latency of < 1ms) by a factor of three. These figures have been internally benchmarked, and verified by actual customer experience.

WAN deployments should aim for latency from client to server of less than 200ms. The amount of hops (physical hardware devices) between client and server should be minimized where possible.

LAN deployments should aim for a latency of less than 1ms.

Sizing Summary

Accurate sizing may be difficult mostly because of poorly defined requirements.

Particular weaknesses in the sizing should be documented to give the responsibility to the requester or clients. A sizing project is not a performance guarantee or a promise that the system will work. Caveats are used to highlight this.

Single/Two/Multi-Tier Installation

The decision to move from a single (monolithic) tier to a two/multi-tier deployment can be influenced by many factors:

- Hardware preferences
- External hosting costs
- Software licensing costs
- Number of users
- Relationships with hardware vendor
- Network topology
- Security
- Performance

Single Tier/Monolithic/Scale-Up installation

The single tier/monolithic/scale-up instllation is the most common type of QAD .NET UI deployment. All components are located on a central server, and extra capacity is added by internall scaling up the hardware components (such as adding/activating extra CPUs or adding extra disk drives or more memory).

This deployment is also the easiest to configure, and has the highest confidence level amongst IT administrators.

The cost for this type of configuration may be higher for the hardware at high user counts, but is often offset by the additional maintenance/software licensing and potential hosting costs that a multi tier approach can incur.

Note that a single tier approach may also require extra effort in securing the software environment.

As long as extra capacity is available, there is no compelling reason to move to a multi-tier architecture for performance reasons.

Two Tier Installation

The most common, and default, type of two tier installation is deployed as follows:

Database Server

Database, Connection Manager Client and QAD 2007.1 code.

All business logic and building of the XML documents occurs on this tier. The connection manager client connects to the database via shared memory.

Application Server

Contains the Tomcat webapp.

It is expected that 15 to 20% of the total environment CPU requirements will occur on this tier. While this may be a cheaper solution in terms of raw hardware costs, there is no compelling performance-related reason to choose this architecture. In most cases the addition of a network delay between the webapp and the connection manager will introduce processing delays.

Network Considerations

An easy way to tell if the network is affecting overall performance is to compare those operations that involve the network with those that do not. If you are running a program that does a considerable amount of remote reads and writes, and it is running slowly, but everything else seems to be running as usual, then it is probably a network problem. Some of the potential network bottlenecks might be caused by the following:

- Client-network interface
- Network bandwidth
- Network topology
- Server network interface
- Server CPU load
- Server memory usage
- Server bandwidth
- Inefficient configuration
- Software architecture

Several tools measure network statistics and give a variety of information, but only part of this information is related to performance tuning. Windows Performance Monitor shows network performance in the Windows environment. With UNIX/Linux, the following commands are useful:

ping
The ping command is useful for determining the status of the network and various foreign hosts, tracking and isolating hardware and software problems, and testing, measuring, and managing networks

FTP
You can use the `ftp` command to send a very large file by using `/dev/zero` as input and `/dev/null` as output. This allows you to transfer a large file without involving disks (which might be a bottleneck) and without having to cache the entire file in memory.

netstat
The `netstat` command is used to show network status.

traceroute
The `traceroute` command is intended for use in network testing, measurement, and management

Ethereal (http://www.ethereal.com/)
Ethereal® is used by network professionals around the world for troubleshooting, analysis, software and protocol development, and education. It has all of the standard features you would expect in a protocol analyzer, and several features not seen in any other product. Its open source license allows talented experts in the networking community to add enhancements. It runs on all popular computing platforms, including UNIX, Linux, and Windows.

Typical QAD .NET UI Bandwidth Requirements

The chart below shows the typical bandwidth requirements of the QAD .NET UI, with comparison to Citrix and character user interfaces.

As can be seen, performing data entry using the cached QAD .NET UI screens takes up virtually the same network bandwith as the Telnet/CHUI interface, and typically less bandwidth than even well tuned Citrix environments.

The areas which do require extra bandwidth are:

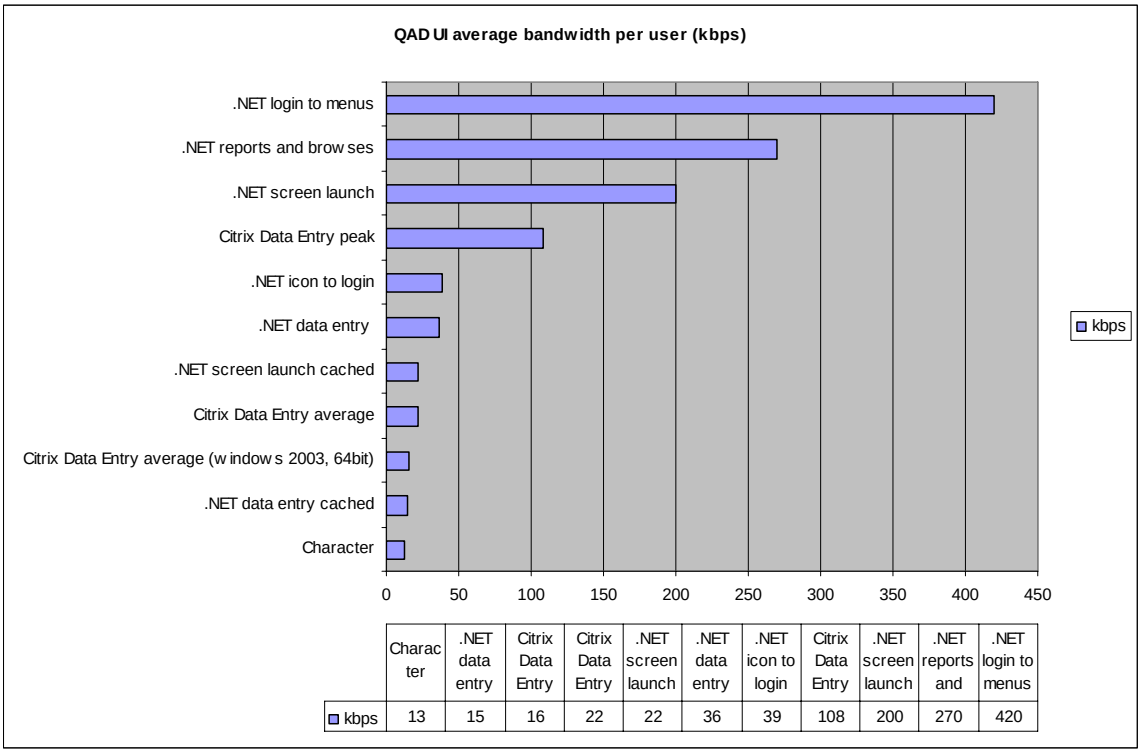
Logins. For the period of login, from the splash screen to the menu presentation, up to 400kbps of bandwidth may be required. This is for a brief (a few seconds) period.

Launching a screen. For approximately a second, there is a short burst of network traffic up to 200 kbps.

Large user reports, browses, and data downloads. These will tend to consume as much bandwidth as they can. The range is typically around 250 kbps, for the duration of the report/browse.

Note that 1 kbps is usually 0.125 kB/second; 400 kbps is 50kB/second.

1 kbps = 0.125 kilobytes / second



	Character	.NET data entry	Citrix Data Entry	Citrix Data Entry	.NET screen launch	.NET data entry	.NET icon to login	Citrix Data Entry	.NET screen launch	.NET reports and	.NET login to menus
kbps	13	15	16	22	22	36	39	108	200	270	420

Client Sizing

Summary

From internal and customer testing done on two client machines (minimum and preferred specification), the responsiveness of the QAD .NET UI interface is much higher on faster machines. The 2.5 GHz client typically responded in about half the time as the 1.5 GHz client. It is expected that clients under 1.5 GHz would continue the trend in slower responsiveness to the point where the application would be unusable in any real sense. Faster clients can achieve exceptionally good screen navigation speeds.

Preferred Client

Intel Core 2 Dup E4400 2.13 Fhz or AMD Athlon 64 2.4 GHz (or better)
1 GB RAM (or more)
1 x 80 GB SATA hard drive (7200 RPM or faster)

Minimum Specification Client

1.5 GHz Pentium IV
512 MB RAM
1 x 40 GB IDE hard drive (7200 RPM)

Disk Usage

Disk usage is minimal (less than 40 MB), and disk activity is also minimal. This interface is not expected to cause disk drive congestion.

Memory Usage

Initial launch of the QAD .NET UI client consumes about 50 MB of system memory.

Memory usage can increase during use of the QAD .NET UI. When dealing with large data sets (in hundreds of thousands of records range), memory use increases rapidly. The QAD .NET UI clients have been seen to consume upwards of 100MB of memory during tests

CHAPTER 5

QAD Internal Benchmark Results

QAD has conducted intensive internal performance testing and benchmarking of the QAD .NET UI. Selected summary information is presented in this section:

Response Times under Load

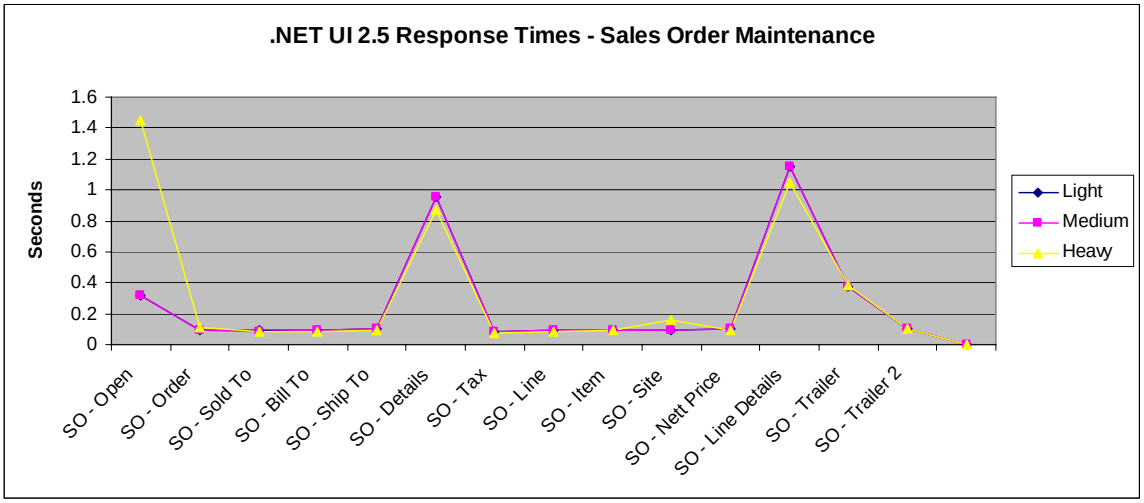
Application Scalability

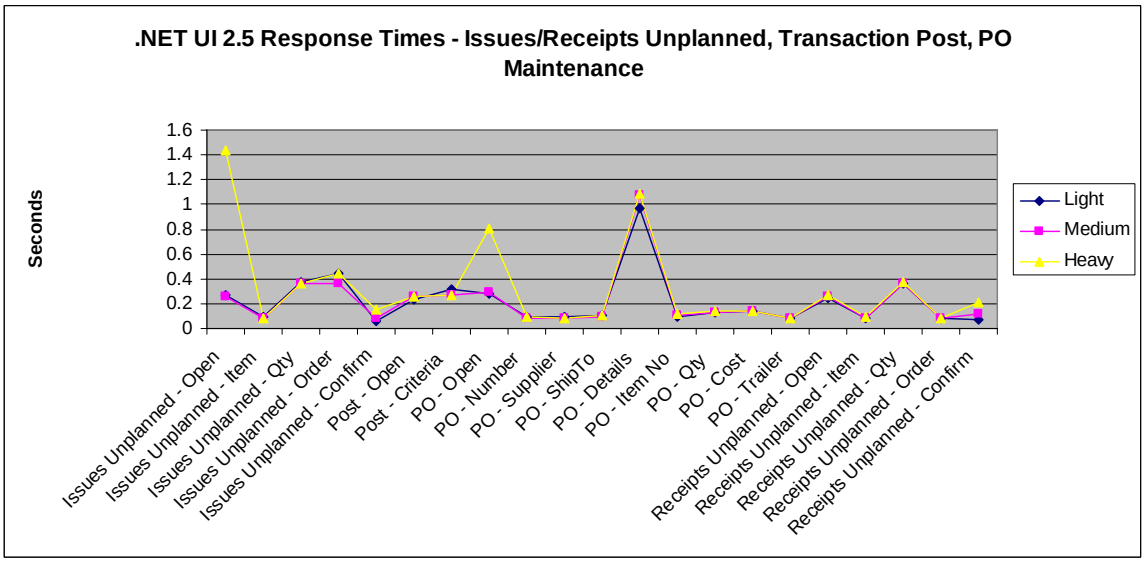
Response Times under Load

The following charts show the field to field response times of selected QAD 2007.1 screens running under the QAD .NET UI 2.5. (The test details are on page 45.)

On a well-tuned system on a fast local network, response times for simple field to field navigation under 200ms is possible even with user counts of 100 or more. Some operations may take longer to complete if the underlying business logic is complex (for example, calculating the tax and freight charges on a trailer record).

Under heavier load, there may be a delay of a second (or slightly more) when launching screens for the first time.





Application Scalability

Screen Navigation

As can be seen from the response times under load, the QAD .NET UI shows extremely good scalability. As the concurrent user count increases, the response times stay very consistent within field to field navigation on the QAD .NET UI screens. The primary indication the system is under heavy load is that *launching* the screens can take longer.

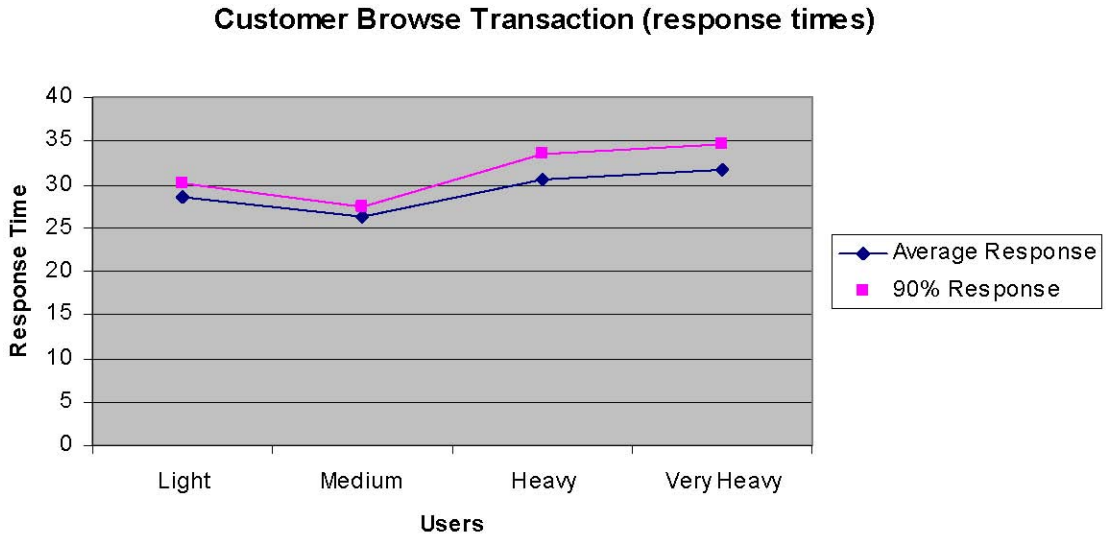
Browses

This test is a black box test driven from a Progress 4GL script. A QAD .NET UI browse (adbr001.p – Customer Browse) was launched against a QAD 2007.1 database containing approximately 45000 customers. The following actions were then carried out:

- Display 1000 records
- Get the total record count (max record count set to 50,000)
- Get next record
- Get previous record
- Get last record

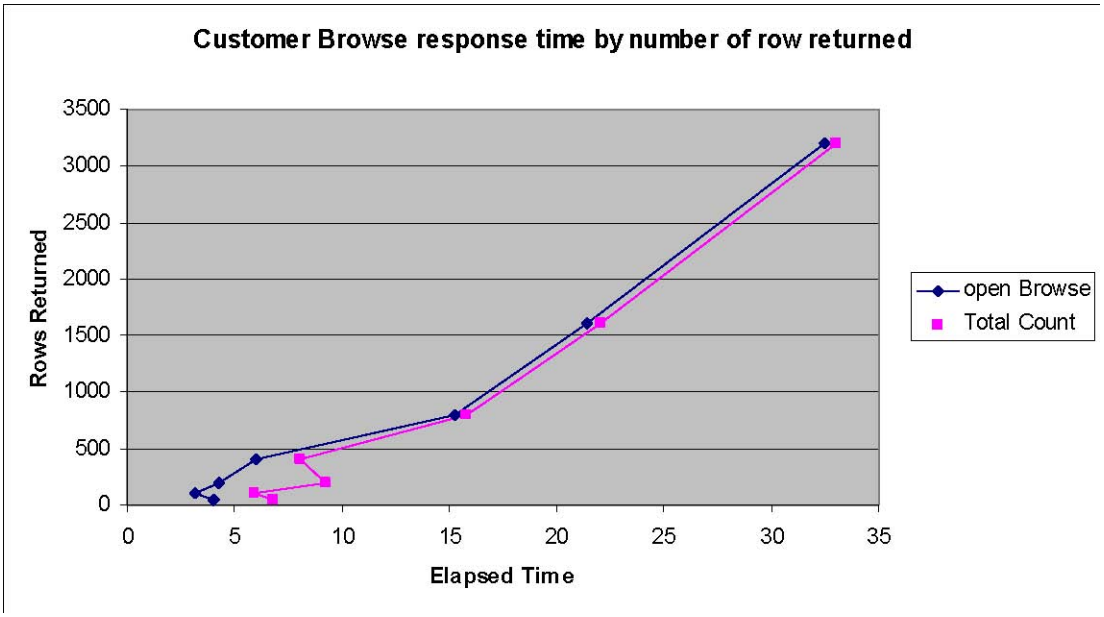
The total response time was recorded. The purpose of this test was to track the scalability of the customer browse under load.

As can be seen from the graph below, response times remain fairly consistent as the user count increases, indicating good scalability.



Response Times of Browse by Number of Rows Returned

The response times for a QAD .NET UI browse show good scalability as the number of rows returned increases. Ignoring low record counts, there is a strong linear correlation between response time and rows returned, as the following graph shows.



CHAPTER 6

Reference

Test Hardware and Software

Test Software

Database Server

2 x dual core 2.8 GHz Intel Xeon CPU (64 bit)

8GB RAM

640GB hard drive space, 5 x Ultra 320 SCSI disks

2 disks in 1 stripe set
3 disks in 1 stripe set

Red Hat Linux Advanced Server release 5 (Tikanga)

Kernel 2.6.18-8.e15 (64 bit)

Summary Tuning information

The Database used for this testing was a real customer database, with approximately 15 GB of data.

Database Block Size : 8192

BI log Cluster Size : 16 MB

BI log Block Size : 16 kB

-bibufs 25 –Mf 6 –spin 50000 –B 20000

2 x Application Page Writers

1 x BI writer

Application Test Software

Item	Vendor	Version
MFG/PRO	QAD Inc.	eB2.1 SP6
Progress Database Engine	Progress Software Corp.	10.1BSP2
Progress WebSpeed transaction server	Progress Software Corp.	10.1BSP2
Progress AppServer	Progress Software Corp.	10.1BSP
Java SDK	Sun Microsystems, Inc.	1.5.0_04
Red Hat Enterprise Linux	Red Hat, Inc.	ES 5 (2.6 kernel)
QAD .NET UI	QAD Inc.	2.5
Tomcat	Apache Software Foundation	5.5.20
LoadRunner load testing software	Mercury Interactive Corporation	8.1.1.1
Test Complete	Automated QA	4.32

Load Test Software

The QAD test execution environment was made up of a number of parts controlled by LoadRunner, a commercial load testing tool from Mercury Interactive Corp.

For testing the browse functionality, an in-house test harness was developed to simulate the traffic for a number of clients, doing simple browse transactions of opening and navigating the browse. A 4GL client was used and driven by a telnet session controlled by LoadRunner. This allowed the full test to be controlled by LoadRunner.

The final traffic generation was a full UI client. QAD used Test Complete by Automated QA for this task.

Test Scenario

This section documents the test scenario used to generate the benchmark results in this document.

A series of 1 hour tests were conducted with the following user counts. Each user was added in ramp intervals of 15 seconds.

Light Test (20 Concurrent Users)

Purchase Order (5 users)

Sales Order Maintenance (2 users)

Issues Unplanned (2 users)

Receipts Unplanned (2 users)

Sales Order Credit Inquiry (3 users)

Transaction Post (1 user)

.Net UI Browses (5 users)

Medium Test (50 Concurrent Users)

Purchase Order (12 users)

Sales Order Maintenance (5 users)

Issues Unplanned (5 users)

Receipts Unplanned (5 users)

Sales Order Credit Inquiry (7 users)

Transaction Post (1 user)

.Net UI Browses (15 users)

Medium-High Test (100 Concurrent Users)

Purchase Order (25 users)

Sales Order Maintenance (10 users)

Issues Unplanned (10 users)

Receipts Unplanned (10 users)

Sales Order Credit Inquiry (15 users)

Transaction Post (1 user)

QAD .Net UI Browses (29 users)

Think Times

Think time for each of the scripts are as follows (in seconds):

Script	Think Times
Purchase Order	205
Sales Order	135
Issues Unplanned	350
Receipts Unplanned	290
Sales Order Credit Inq	172
QAD .Net UI Browses	200