Microsoft

Enterprise Library for .NET Framework 2.0: Performance Comparison 64 Bit vs. 32 Bit

Authors: Larry Brader, Carlos Farre, Pavan Kumar Sura (Volt)

patterns & practices

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2006 Microsoft Corporation. All rights reserved.

Microsoft, MS-DOS, Windows, Windows NT, Windows Server, Visual Studio are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Table of Contents

Analysis of Enterprise Library

The patterns & practices Enterprise Library is a collection of application blocks designed to assist developers with common enterprise development challenges. Application blocks are a type of guidance, provided as source code that can be used "as is," extended, or modified by developers to use on enterprise development projects. To find out more about Enterprise Library and to obtain a copy, visit Enterprise Library page on MSDN, or the Enterprise Library Community site.

Caveat

Performance results are very specific and can vary depending on the configuration of hardware and software environments, **Enterprise Library customers should expect different results for their own applications.**

The Analysis

The following analysis is of the Enterprise Library deployment in both a 32-bit (2 and 4 processor) and 64-bit (2 and 4 processor) environment and examines their relative performance to a 32-bit 2 processor server. To get a realistic comparison to measure since the 32-bit and 64-bit worlds have differences, the setups used were designed to be as equivalent as possible. The goal was to obtain the relative performance measurements of the Enterprise Library Logging Application Block, Caching Application Block, Data Access Application Block and the scalability from going from a 2 to a 4 bit processor arrangement in each environment. The various configurations and measurement results are below.

Some Fast Answers

- Q: How long did it take to convert Enterprise Library from 32 bit to 64 bit?

  A: None, since the .NET Framework uses an intermediate language which is processor independent and uses a just-in-time compiler to convert it to machine code as required by the deployment environment. So it's not a recompile—rather just run it on a 64-bit .NET-based machine—although, there can be specific issues see the section "Moving to the 64-Bit World."
- Q: Does the Enterprise Library scale?

  A: Yes, the results show a 50 to 70 % gain in the tested scenarios going from a 2 to 4 processor environment regardless if it's 32 bit or 64 bit, see section 4 for details.
- Q: What are the gains in going 64 bit?

  A: Scabililty, see section "Scabililty View of the Caching Block" for details

Before looking at the full analysis, let's first look at how it was done, and check out some resources to get more information on the 64-bit solutions.

Moving to the 64-Bit World

Historically the moves from 8-bit processors to 16-bit processors and 16-bit processors to 32-bit processors were big jumps for the industry and customers. The 64-bit architecture changeover is quiet in comparison.

This is an ongoing evolution—and the era of the 32-bit processor is closing out.

Resources for Migrating to 64 Bit

The following Web sites can help answer your 64-bit migration questions.

- .NET Framework Developer Center: 64-Bit .NET Framework
- Windows Hardware Developer Central: 64-bit System Design
- MSDN Library Development Guides: 64-Bit Windows
- 64-Bit Computing Advantage Microsoft & Intel
- x64 Primer: Everything You Need To Know To Start Programming 64-Bit Windows Systems

Test Deployment Configurations

In order to do a relative performance comparison, the following server and client configurations were used to gather the required performance measurements. All of this is dependent on the performance tools being used, in this case we used the Performance tool set that comes with Visual Studio Team System.

Visual Studio Team System 2005—Performance Tool

Visual Studio Team System 2005 is a very powerful development system, and included in the latest release is a performance tool which will handle the measurements required to conduct performance tests. The main components and their functions are described below:

1. **Visual Studio**: Used to generate scripts that simulate the real time Web access through various configurable parameters including the number of concurrent virtual users.
2. **Controller**: Used to start the tests and also read though the different performance counters and graphs while the test is running. .
3. **Agent**: Used to generate the load, installed on each client.

More information about Visual Studio Team System 2005 and can be found at Software Architect Team Center site.

Server Configurations

The server configurations used in these tests are listed in Table 1.

Table 1. Server Configurations Details

	Server A	Server B	Server C	Server D	Server E
Architecture	32 Bit	32 Bit	64 Bit	64 Bit	64 Bit
# Processors	2	4	2	4	4
Processor Speed	2790 MHz	1903 MHz	1804 MHz	1804 MHz	1804 MHz
Physical Memory	1024 MB	4048 MB	3072 MB	3072 MB	5098 MB

Note: The 32-bit and 64-bit memory configurations are different because 32 bit and 64 bit use the RAM memory differently, so having them be identical wouldn't be a meaningful test. Rather understanding how the RAM is used for a 32-bit versus 64-bit deployment is more important.

Note: All 64-bit systems ran on the 64-bit version of the operating system.

Characteristics of Processors:

- **Server A**: 32 Bit processors, 2790 MHz, with a 512 KB level 3 cache and Hyper Threading technology.
- **Server B:** 32 Bit processors, 1903 MHz, with a 2 MB level 3 Processor cache and Hyper Threading technology.
- **Server C/D/E**: 64 BIT processors, 1804 MHz, with a 1 MB level 3 Processor cache and Hyper Threading technology.

Client Configurations

The configuration details of the clients used to gather the performance measurements for each test are listed in Table 2.

Table 2. Client Configuration Details

	Test Client A (Visual Studio agent Clients)	Test Client B	Test Client C
Machine Name	NPSCODECLNT05	NPSCODECLNT01	NPSCODECLNT02
Architecture	32 Bit	32 Bit	32 Bit
Number of Processors	2	1	1
Processor Speed	2787 MHz	1396 MHz	1396 MHz
Total Physical Memory	1024 MB	1024 MB	1024 MB
Available Physical Memory	524 MB	574 MB	664 MB

Analysis Test Scenarios

The Enterprise Library for .NET Framework 1.1 (June 2005 release) test cases were used with only minor changes to standardize the tests for the target servers. As mentioned, performance results will vary with hardware, software configuration, and application. A user is encouraged to examine their performance for their application.

The test scenarios are broadly classified into the following three groups which are a subset of the six application blocks forming the Enterprise Library.

- Logging Application Block. A message is written to the Event Log.
- Caching Application Block. The cache item is represented by a key value pair. A random key item is added to the memory implementation cache from a pool of range values for the keys. The agents use a set of defined keys, from a pool of defined keys. Since the ranges of the keys are larger than the virtual users, executing the load test, at any given moment they are unique per load test execution. When the agents use all the possible values from the pool, then the test recycles setting the starting point to the beginning of key set. This way the test also guarantees a finite number of key values allocated to the agents.
- Data Access Application Block. Execution of the 1 stored procedure returning 1 row from the products Northwind database.

Table 3. Test Scenarios

	Test Scenario Name	Number of Virtual Users
1	Logging Writing to Event Sink	1,10,50,150
2	Caching Adding Items In Memory	1,10,50,150

	Cache Manager	
3	Data Block Execute Non Query	1,10,50,150

Performance Test Goal

Test scope for Performance and Stress aiming at validation of following items:

- **Consistency.** The results of transactions are consistent during performance and stress, and the updates are completed.
- **Availability.** Application and machine are always available. Exceptions are handled accordingly—no side effects, interruptions of service are handled by the framework. Testing done over a minimum 24 hour test cycle.
- **Performance.** Response times for the transactions along with system resources monitoring:
    - Processor
    - Memory
    - Disk
    - Common language runtime (CLR) memory and CLR locks and threads.
- **System Resources.** Memory/Thread consumption, Network, System devices.
- **Optimization, Critical Code Paths.** Indication of expensive code paths pinpointing optimization alternatives.
- **Indicative of Performance Problems.** Indication of problems at the performance counters and function calls level—including initialization costs, access to configuration data, runtime function calls, and stress test behavior.
- **Stress Testing.** Application blocks DLL code tested over a minimum 24 hour test cycle.
- **Scalability Testing.** Throughput measured on 32-bit and 64-bit 2 and 4 processors systems.

Performance Test Methodology

The following lists the process followed to obtain the performance measurements, all tests were done on Visual Studio Team System 2005.

1. Test Harness was built with a ASP.NET wrapper hosted by a Web server.
2. Client Test harnesses were built with Visual Studio Team System. Tests developed in Visual C#.
3. The thread pool used is from IIS infrastructure. On the client side the virtual user is spawned per thread.
4. The thread pool was set as recommended by Chapter 17 of Improving .NET Application Performance and Scalability.
    a. MaxioThreads at 100
    b. MaxWorkerThreads at 100
    c. MinFreeThreads set to 88 * # CPUs
    d. MinLocalFreeThreads set to 76 * # CPUs
    e. MaxConnection set to 12 * # CPUs (note: no effect on Enterprise Library)
    f. Stress testing over 72 hours

   g. Profilers (Icecap and office profiler)

5. Four client load machines were used to avoid system resources maxing out with shorter transaction times.
6. First hit to the server page was cached to avoid the initialization cost.
7. Randomization of message sizes/cached keys/encryption providers was adopted with 50bytes – 3000bytes from the load engine.

Logging Application Block Scenario: Write to Event Log

For this test scenario, the machine configurations were deployed as shown in Table 1. The results of the test are shown in the following graphs; note that the performance has been normalized to gauge, having the 32-bit 2 processor system as the baseline processor system.

Response Time

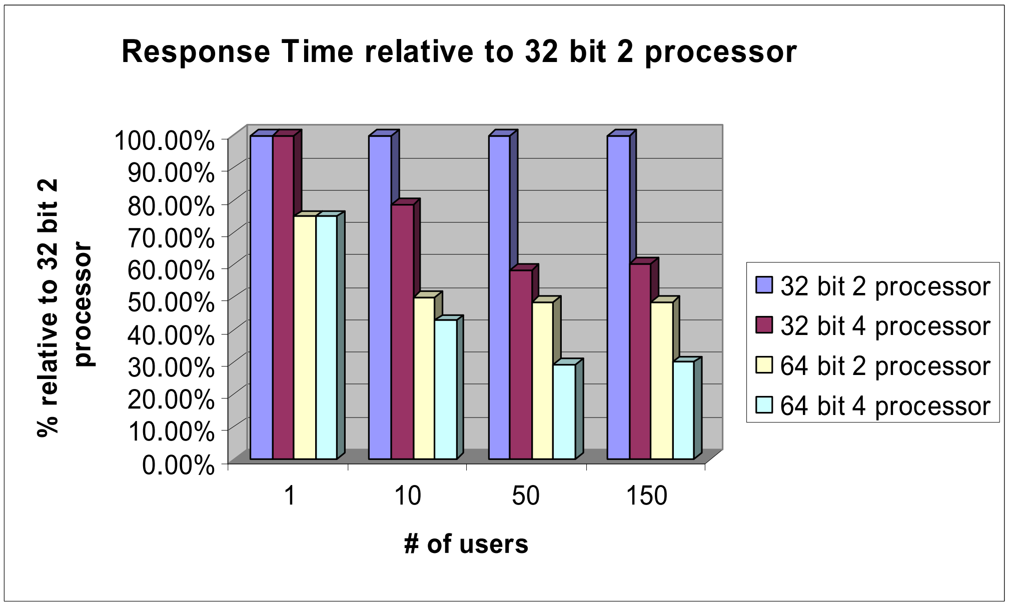

Figure 1, Logging Block response time relative percentage to 32-bit 2 processor setup

Transaction per Second

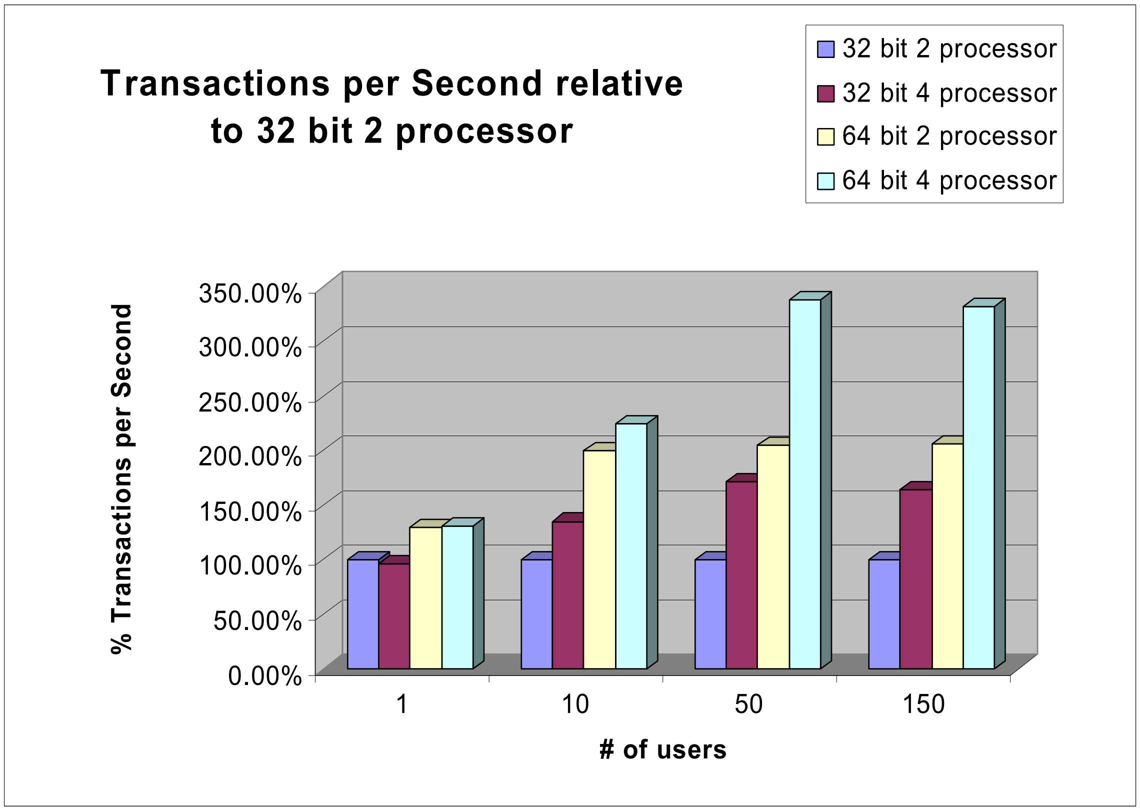

Figure 2, Logging Block transactions per second relative percentage to 32-bit 2 processor setup

Percentage of Processor Time

The processor time graph below shows the percentage of processor time is being used for each processor setup for each use case with an increasing number of users.

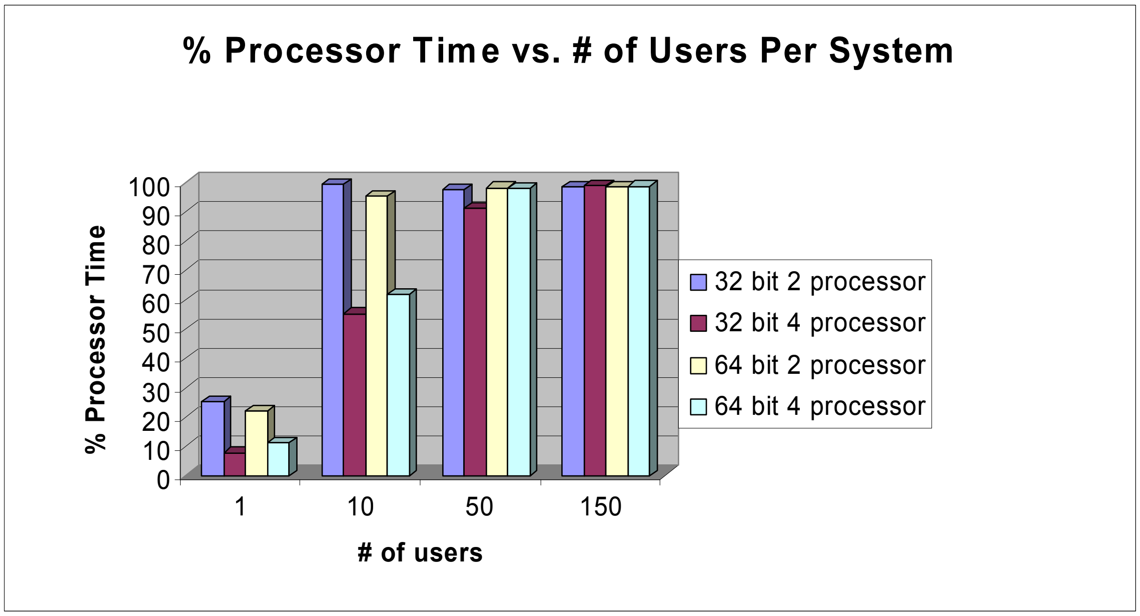

Figure 3, Logging Block percentage of processor time

Test Notes

Following are notes observed during the above performance tests.

- The server machines were always running at 100 percent with no other dependency in order to characterize the system bottlenecks besides processor constraints.
- Memory did not play an important factor in determining data variation from 2 to 4 processors for both 32-and 64-bit machine setups.
- 64-bit machines out performed their 32-bit counterparts (2-4 processors) by 45% on transactions per second and total transactions.
- There was no IO activity observed with percentage of disk time no higher than 6% on all cases.
- The turning point occurred when there was no capacity left for more concurrent processing:
  - 32 bit 2 processors at 10 users
  - 32 bit 4 processors at 50 users.
  - 64 bit 2 processors at 10 users.
  - 64 bit 4 processors at 50 users.
- The gains in scalability increased as numbers of concurrent users were added to the test mix, bigger deltas were observed between 2-4 processors (32 bit and 64 bit on the same and different platforms). The scalability of the logging block is considered good because of the low contention rate per second as the load was increased.
- Measurements from w3wp.exe, a worker process for ASP.NET, created a working set and common language runtime memory allocation that indicated there were no memory pressures on the machines during load test with only the CPU being the bottleneck.

- Percentage of contention rate per second was around 8 with saturation of CPU in all cases and 2-3% below saturation levels.
- Working sets of the host were below 40 MB across all test cases, in all platforms.
- Processor queue lengths started increasing with CPU bottleneck which was caused due to the number of concurrent users. The queue length values begin with zero and increase to seven to ten at the tipping point level.

Data Access Application Block Scenario: ExecuteNonQuery

ExecuteNonQuery was used in a read only scenario for the 64-bit testing and the following machine configurations were deployed as shown in Table 4.

Table 4. System setup for ExecuteNonQuery

	Server A	Server B	Server C	Server D
Architecture	32 Bit	32 Bit	64 Bit	64 Bit
# Processors	2	4	2	4
Processor Speed	2790 MHz	1903 MHz	1804 MHz	1804 MHz
Physical Memory	1024 MB	4048 MB	3072 MB	3072 MB

Response Time

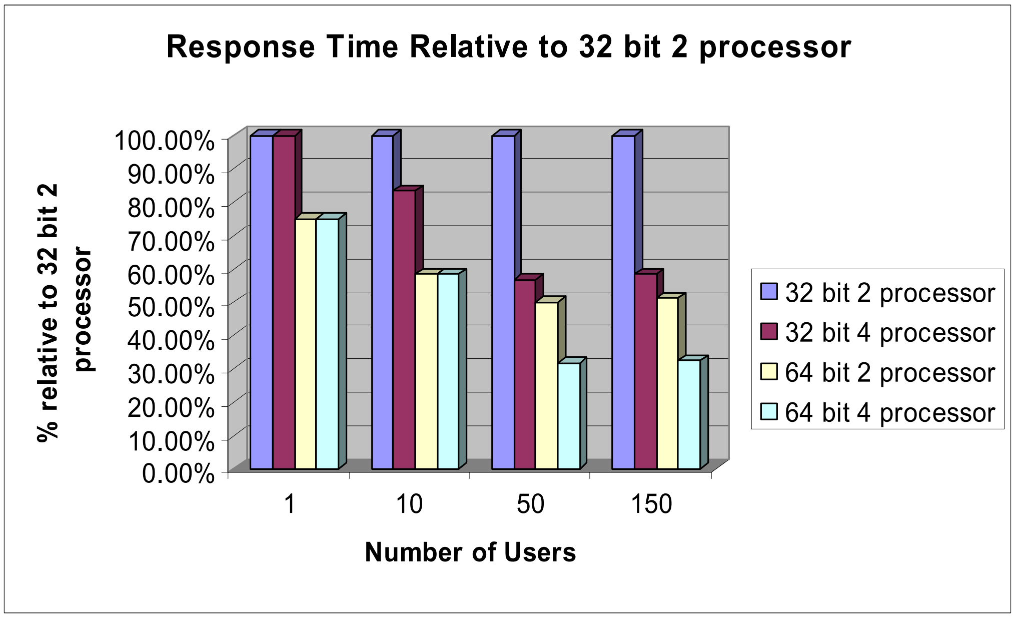

Figure 4, Data Access Block response time relative percentage to 32-bit 2 processor setup

Total Transactions

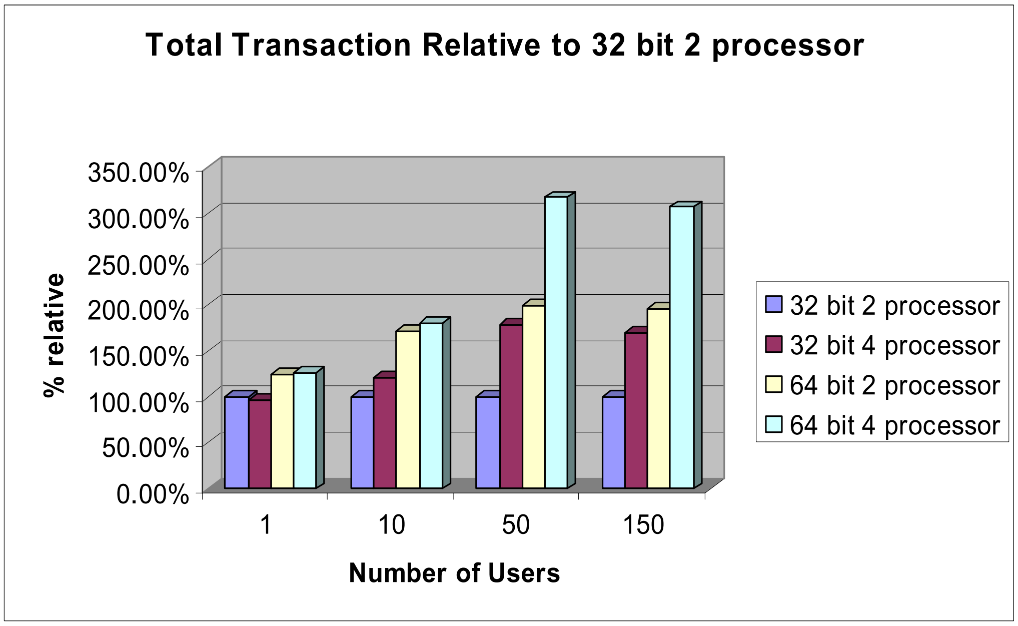

Figure 5, Data Access Block total transaction relative percentage to 32-bit 2 processor setup

Transactions per Second

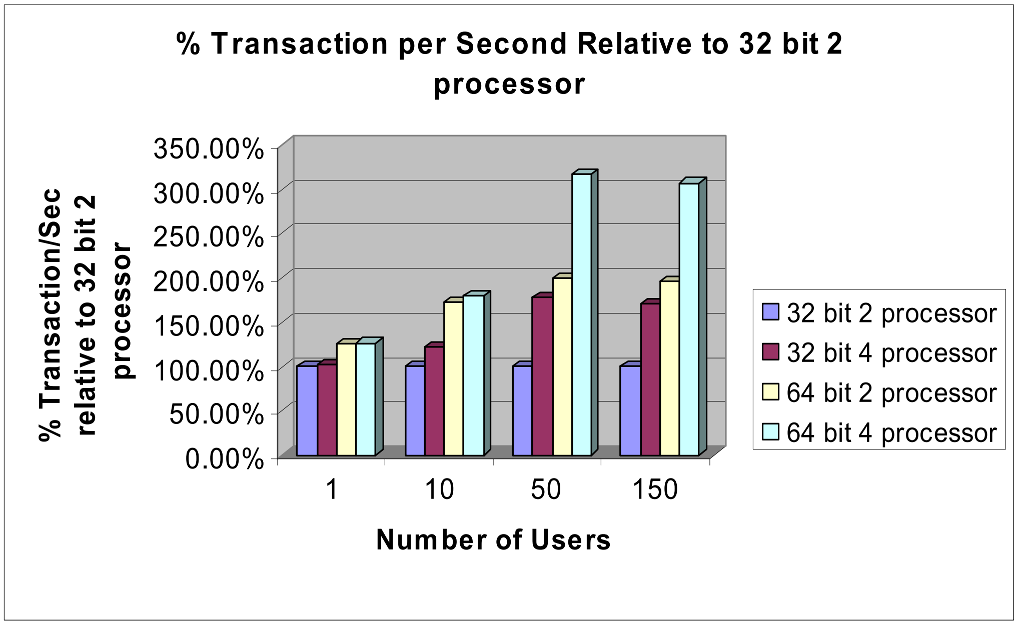

Figure 6, Data Access Block Transactions per second relative percentage to 32-bit 2 processor setup

% of Processor Time

The processor time graph below shows the percentage of processor time that is being used for each processor for each use case with an increasing number of users.

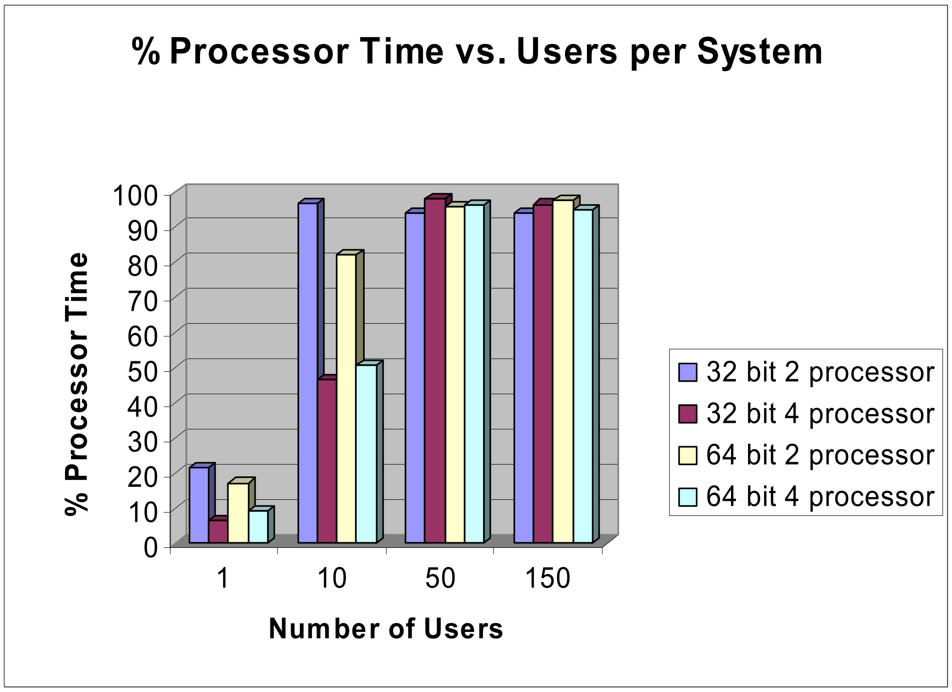

Figure 7, Data Access Block %Processor time with number of users relative percentage to 32-bit 2 processor setup

Scalability View of the Data Block

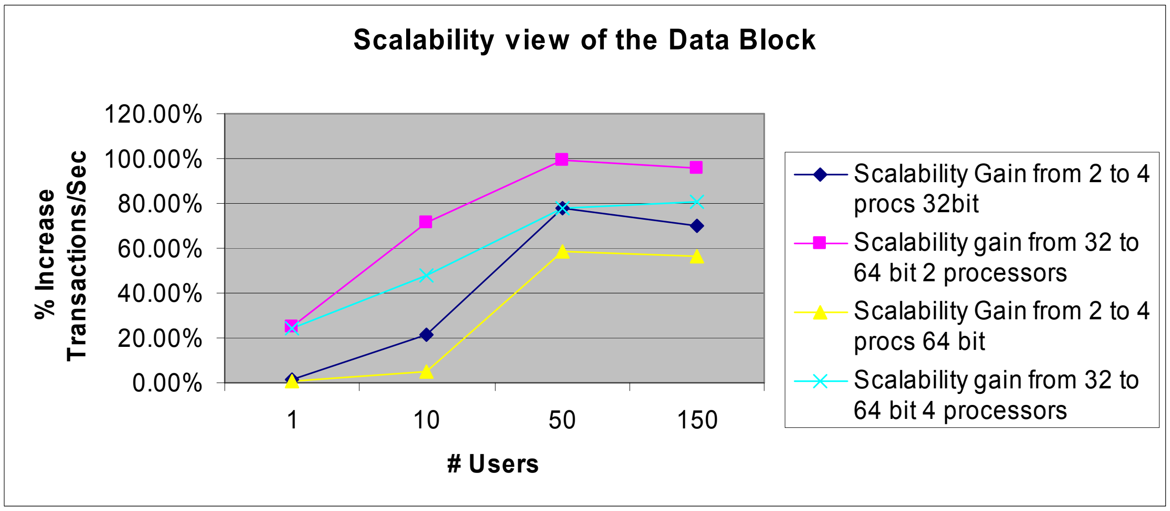

Figure 8, Scabililty view of the Data Access Block

Test Notes

Following notes observed during the above performance tests.

- The server machines were always running at 100 percent with no other dependency to characterize system bottleneck besides the processor constraints.
- Memory did not play an important factor in determining data variation from 2 to 4 processors for both the 32 and 64 bit machine setups.
- 64-bit machines out performed their 32-bit counterparts (2-4 processors) by 45% on transactions per second and total transactions.
- There was no IO activity observed with percentage of disk time no higher than 6% on all cases.
- The turning point occurred when there was no capacity left for more concurrent processing:
  - 32 bit 2 processors at 10 users
  - 32 bit 4 processors at 50 users.
  - 64 bit 2 processors at 50 users.
  - 64 bit 4 processors at 50 users.
- A good sign of scabililty is shown by the performance gains as the numbers of concurrent users were added to the test mix, note that bigger deltas were observed between the 2-4 processors setups for both the 32-bit and 64-bit configurations.
- The scalability of the logging block is considered good because of the low contention rate per second as the load was increased.
- Measurements from working sets and activity of percentage of the GC indicated that there were no memory pressures on the machines during load test with only the CPU being the bottleneck.

- Percentage of contention rate per second was around 8 with saturation of CPU in all cases, and 2-3% below saturation levels.
- Working sets of the host were below 40 MB across all test cases in all platforms.
- Processor queue lengths start increasing with the CPU bottleneck caused by the number of concurrent users. Values begin with zero and increase to seven to ten at the tipping point level.
- To guarantee that CPU was going to bottleneck, the server machines read one row of the database for every test cycle. Also, no writes are submitted because it could cause the SQL Server to increase IO activity. The purpose of the procedure was to avoid any dependency that would halt CPU consumption at its full capacity.

Caching Block Scenario: Adding Random Item in the Cache in Memory

For this test scenario, the following machine configurations were deployed as shown in Table 5.

Table 5. Caching Block Scenario

	Server A	Server D
Architecture	32 Bit	64 Bit
# Processors	4	4
Processor Speed	1903 MHz	1804 MHz
Physical Memory	4048 MB	5098 MB

Response Time

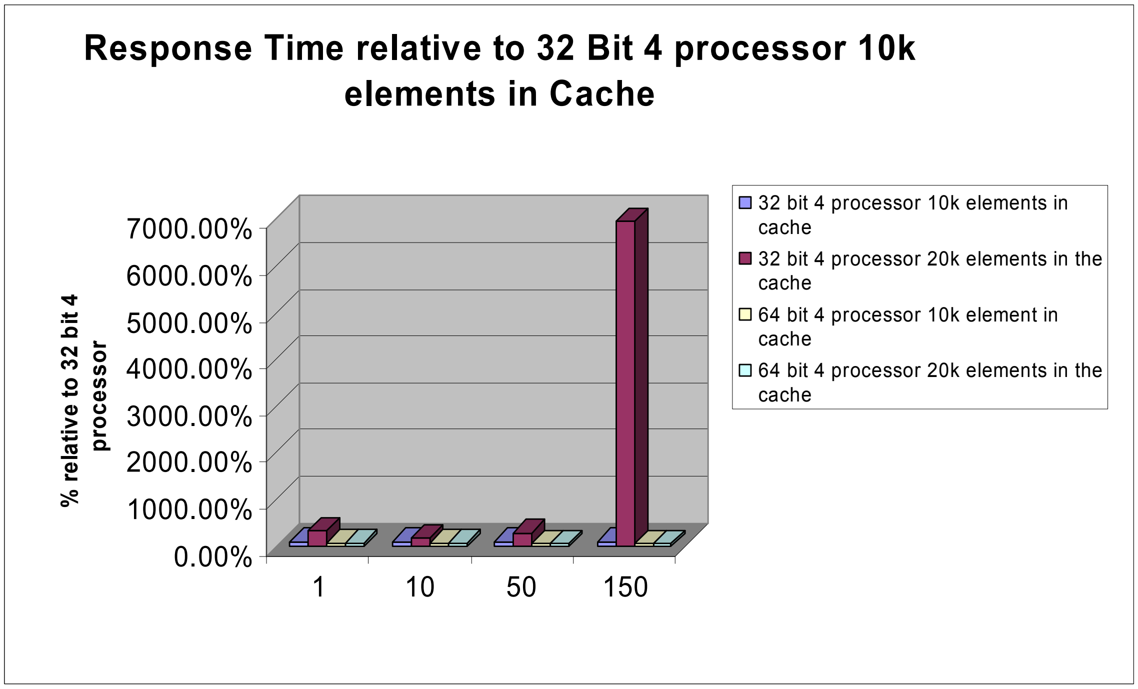

Figure 9, Caching Block response time relative percentage to 32-bit 4 processor with 10 thousand elements in cache

For this scenario, as shown in Figure 9, the performance test stressed the 32-bit 4 processor configuration to show the working set point of saturation by inserting 10 and 20 thousand elements in the cache. Meanwhile, this same working set is no problem with the 64-bit systems due to their address space.

This spike shows the saturation point, it very hard to compare the relative performance to the 64-bit system. Figure 10 is redone with the 32-bit 4 processor 20 thousand elements removed to show the relative performance difference to the 32-bit 4 processor with 10 thousand elements.

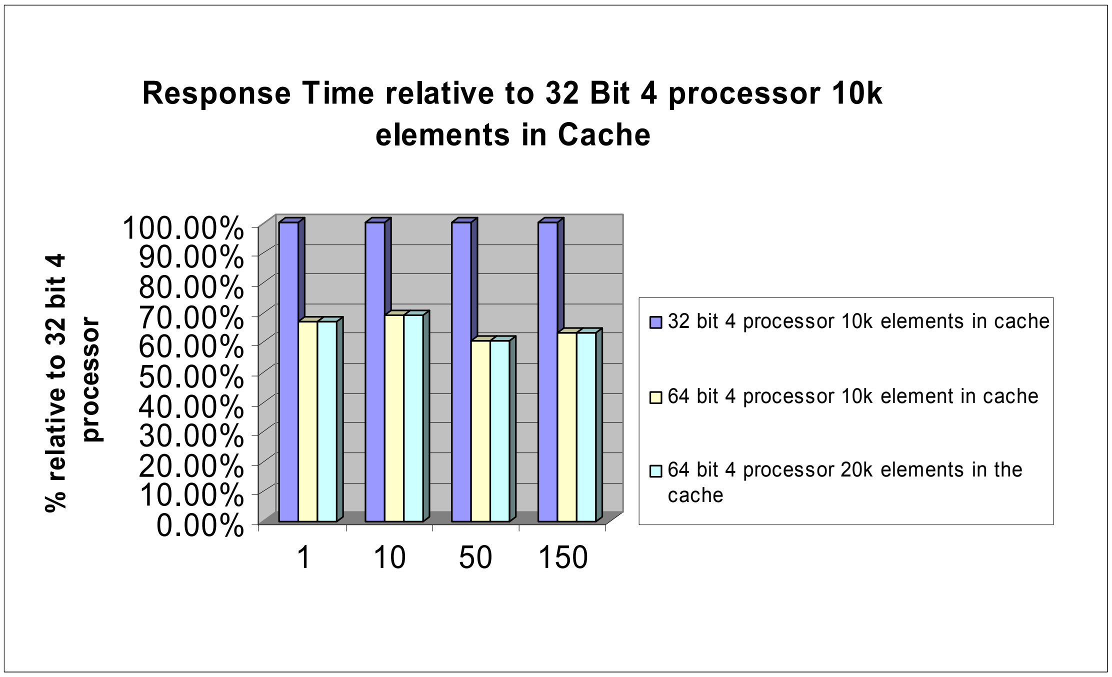

Figure 10, Caching Block response time relative percentage to 32-bit 4 processor with 10k elements in cache with no 32 bit 4 processor, see figure 9 to see view the performance comparison with the 32 bit 4 processor.

Transactions per Second

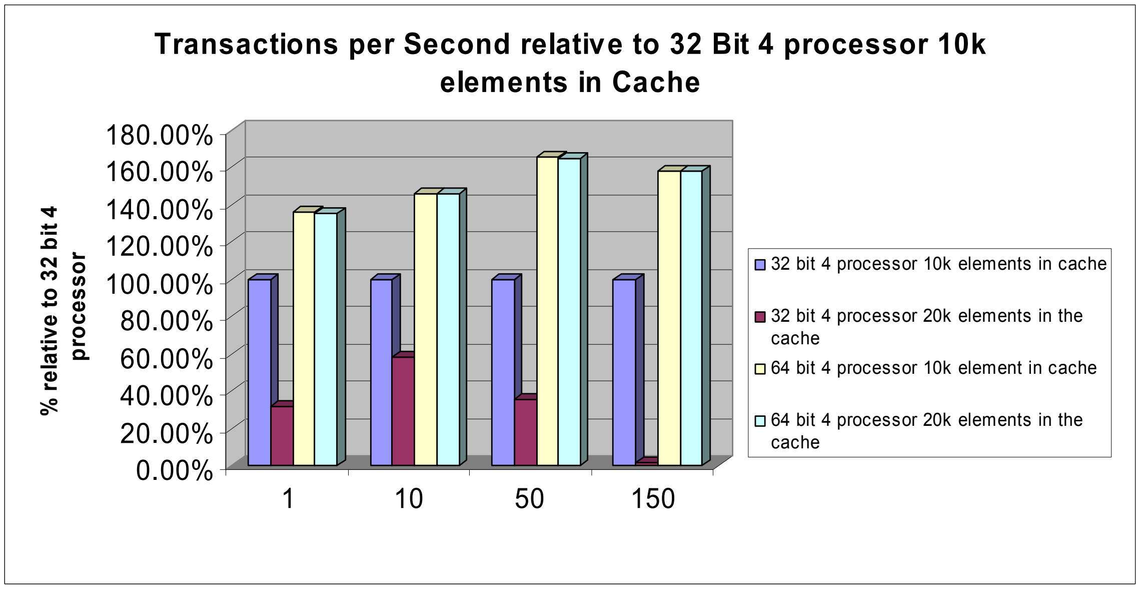

Figure 11, Transactions per second relative to 32-bit – 4 processor 10 thousand elements in cache

Percentage of Processor Time

The processor time graph in Figure 12 shows the how much work a processor is doing per each use case. As before the graph data is normalized and relative to the 32-bit 2 processor system, as shown this system is set to 100%. In the earlier Logging and Data Access tests, the processor time of the 32-bit – 4 processor and 64 bit 2 & 4 processor consumed less processor time then the 32-bit – 2 processor system. Only as the load increased did the processor time for all systems go to 100% usage.

What's interesting in this caching scenario is that the 32-bit – 4 processor and 64-bit 2 & 4 processor actually consume more processing time then the 32-bit – 2 processor, although the throughput is also higher then the 32-bit – 2 processor system.

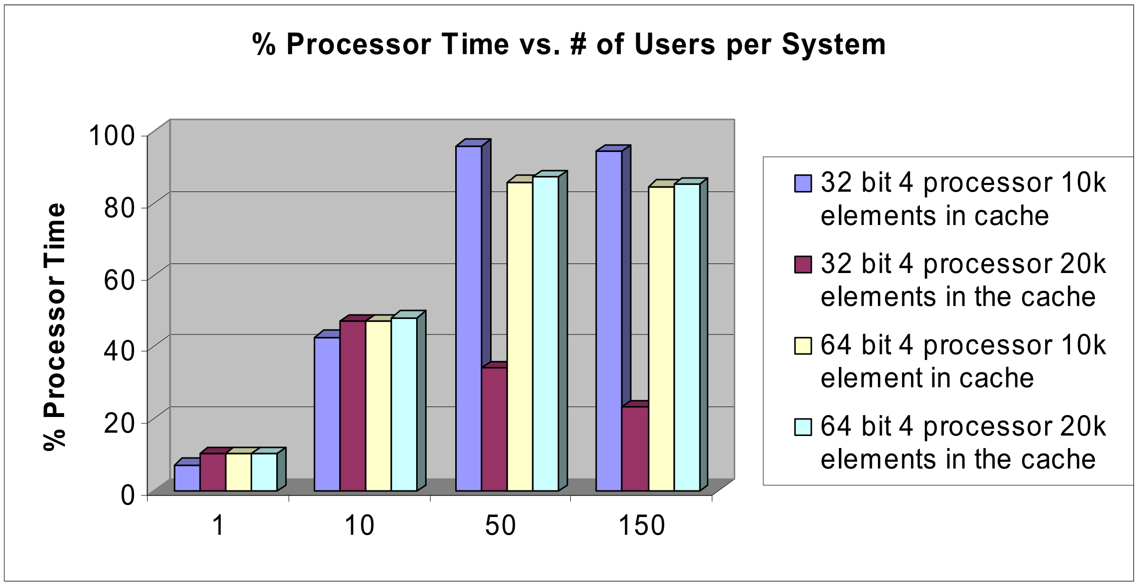

Figure 12, Percentage of processor time versus. Number of users per system

Scabililty View of the Caching Block

Figure 13 shows the scalability that occurs within the Caching Block, There are three lines, two show the scalability effect observed for the 32-bit and 64-bit configuration, while the third is a percentage comparison between the scalability gained. The scalability effect is observed by noting the percentage increase in transactions per seconds as the number of users are increased or stay the same. In the 32-bit configuration, the percentage of transactions per second decreased, while the 64 bit had no change, indicating that the increased load had no effect. The scalability gain then for working in the 64-bit configuration versus the 32-bit configuration can be observed in the third line.

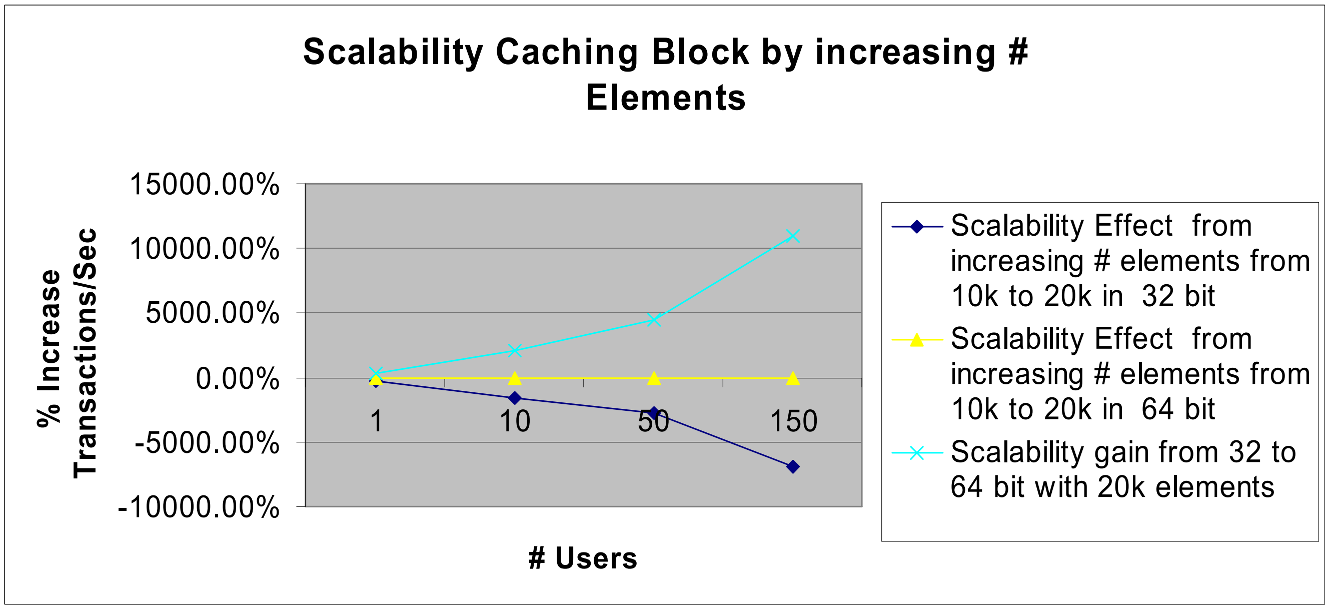

Figure 13, Scabililty view of the Caching Block

Test Notes

The Caching Block test consists of inserting elements in the cache with in-memory implementation. The caching item is identified as a key-value pair. The test harness generates a random key value from 1 to 20000 with a payload of 100 KB. The 64-Bit common language runtime is only able to allocate 1 GB of size to a given object, but it is possible to allocate as many objects below 1 GB in size provided there is memory available. So the tests cases are divided by inserting 10 thousand and 20 thousand items with payload of 100 thousand elements in the cache using 4 processor machines in both 32-bit and 64-bit processor. The test may repeat the same key element in two different transactions. The purpose is to test the scalability of address space for both platforms (32 and 64 bit) machines.

- For the 10 thousand elements example, when it is possible to allocate the elements in the 32-bit address space the gain moving to the 64-bit platform is 30% to 60% when the amount of memory available will dictate the scalability benefit.
- At peak it is possible to use full capacity of the server around 98% for both 32 and 64-bit machines.
- With 20 thousand elements, the 32-bit machine starts paging with no sufficient memory available to hold all the data. Memory / Pages per second counter goes up 160-170 pages per second and processor usage falls to 34% range, because of paging. The private bytes for the host are close to 1.7 GB. On the 64-bit machine it is possible to allocate up to 30 thousand elements sustaining full capacity usage and throughput. At this point private bytes are around 5 GB with low contention and no Pages per second.

Summary

The following is a summary of key observations made from analysis of the test case results. The main relative performance measurements of all the test cases presented in above mentioned test cases should also help the reader to understand the rationale behind some of the conclusions below.

- In all cases there were no external dependencies to affect latency times.
- All scenarios were running at 100 percent, except the case when 20 thousand elements were inserted in the cache using a 32-bit machine. At this point paging is causing IO dictating a dependency without full capacity of machine being used.
- In those cases there were no other system constrains such as memory or disk IO.
- The maximum bandwidth use across all results was 243,500,000 bytes/sec, with the exception of the caching scenario where 5,475,000,000 bytes/sec were being transferred. This represents 40% maximum utilization of network bandwidth only for the Caching block.
- There is no contention in 64-bit or 32-bit machines across all load conditions; this is an indication of scalability for Enterprise Library. Adding more processors and/or memory to the server would result in higher throughput compared with the same number of processors or memory in 32-bit machine. We believe that the scalability is due to less memory latency, bus contention, decreased procedure call overhead, and 64-bit memory addressing.

After going through the numbers from the test results gathered, it can be observed that performance was largely dictated by the number of processors as well as processor speeds on the server.

The 64-bit machine has 2 & 4 processors compared to 2 & 4 processors on the 32-bit machines. Also, the processor speed on 64-bit machines is 1804 MHz per processor as compared to 2790 MHz per processor on 32-bit machines. With these performance numbers in perspective, it can be observed that with less CPU processor power the gains on scalability moving to 64 bit was 100%.

The second important factor for nominal performance of a 64-bit machine revolves around the use of system resource and memory space. For this analysis, we used the Enterprise Library Application Block default test cases. These test cases are not specifically targeted to assess the new features of 64-bit architecture. Since one of the main benefits of 64-bit machine is its high memory and improved architecture, this would be good future work.

Future Work

Potential future performance analysis:

- Conduct scalability testing over larger address spaces.
- Conduct scalability testing with a higher number of processors.
- Confirm that 64-bit common language runtime improves performance.
- Conduct performance tests for parallelism and branching.