RECURSION SOFTWARE, Inc.

Software for Software Developers™

VOYAGER®

High Volume Transaction Processing in Enterprise Applications

By Thomas Wheeler

Recursion Software, Inc.
February 16, 2005

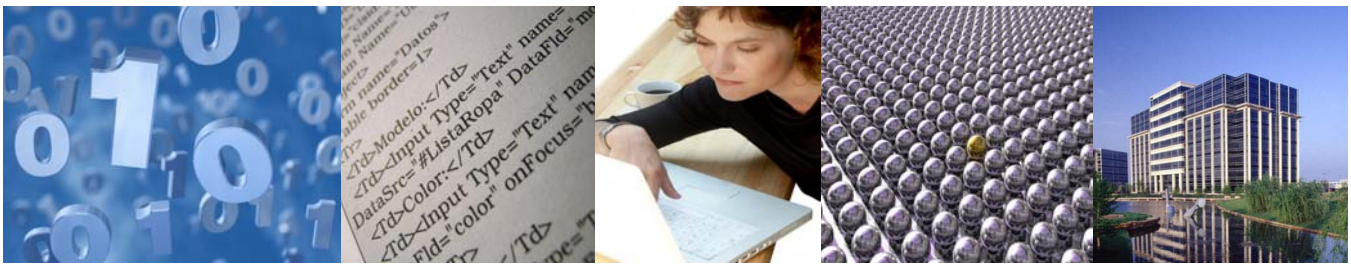

TABLE OF CONTENTS

VOYAGER®

High Volume Transaction Processing in Enterprise Applications

By Thomas Wheeler

Overview

Performance and scalability continue to be critical factors in the development and deployment of enterprise software. Web Services technologies require more processing power and bandwidth than binary protocols due to the use of XML as an underlying encoding. At Recursion Software, we understand that today's implementation of Web Services may not provide the performance and scalability some systems demand. As part of our ***Adaptive Technology Approach***, our products support a binary protocol in addition to the standard Web Services protocols. This paper documents the performance and scalability of Voyager's binary protocol (Release 4.8).

Products, Tools, and Environment

The test environment consists of the following systems:

OS and hardware environment

System 1:
- Model: HP Proliant DL145
- CPU: single processor 1.80 GHz AMD Opteron
- RAM: 4096
- OS: Red Hat Enterprise Linux WS release 3 (64Bit)

System 2:
- Model: Sun-Fire 280R
- CPU: Dual Processor (900 MHz)
- RAM: 4096
- OS: Solaris 10 OS (64Bit)

System 3:
- Model: HP Proliant DL145
- CPU: single processor 1.80 GHz AMD Opteron
- RAM: 2048
- OS: Windows 2000 (32Bit)

System 4:
- Model: HP Proliant DL145
- CPU: single processor 1.80 GHz AMD Opteron
- RAM: 2048
- OS: Red Hat Linux release 9 (32Bit)

System 5:
Model: PC generic
CPU: single processor AMD Athlon 3200 CPU
RAM: 1024
OS: Windows 2000 (32Bit)

Network environment

The network environment is 100MB Ethernet.

Products

Voyager 4.8
Axis 1.1

Methodology

All systems involved in each test were idle prior to starting the test. The processes required for the test were started in the appropriate order. Once started, several minutes were required to allow the processes to stabilize. Benchmarking information was measured over a period of a minute to obtain an average used as the final result.

Performance Benchmark Results

Remote Invocation Performance

The performance benchmark consisted of a single client thread repeatedly calling a method on a server object, passing a 100 byte array as a parameter (Voyager) and a similar-sized string (Axis).

This table shows Voyager's performance (using a binary protocol) to be greater than Axis (using SOAP) by a factor of 16+.

Product	Load	TPS	Delay	Threads	Clients
Voyager	30%	4091	0	1	1
Axis	20%	246	0	1	1

Platform: The product being tested.
Load: The average CPU load on the server.
TPS: The average invocations per second
Delay: The delay between invocations for each client thread
Threads: The number of threads per client process.
Clients**:** The number of client processes.

Remote Invocation Scalability

The scalability benchmark consisted of multiple clients running multiple threads, each thread repeatedly calling a method on a server object. A delay of 25 milliseconds between each client thread invocation is used.

In this test, Axis was unable to scale beyond three clients each with 50 threads. Voyager successfully scaled to a total of 1000 client threads and an average rate of 6518 invocations per second.

Product	Load	TPS	Delay	Threads	Clients
Voyager	100%	6518	25	250	4
Axis	80%	675	25	50	3

Messaging Scalability

Voyager provides a unique, highly scalable message passing feature called *Space*. A Space consists of two or more interconnected *Subspace*s. An event published to one of these Subspaces is broadcast to the entire Space, providing a rapid and scalable event publishing capability. Interconnects between Subspaces are bi-directional.

The messaging scalability tests consist of a Server which acts as the hub of the Space topology and publish events to the Space; Senders, which publish events to the Space and act as clients; and Clients, which act as pure clients, only receiving events.

Three benchmarks were run. In the first, the server and two senders publish events to the Space, and two clients act as receivers. This represents a system with multiple processes both publishing and receiving events. The second benchmark represents a single server broadcasting events to several clients. The third benchmark consists of a server that does not publish events, two senders, and several clients.

These benchmarks show that Voyager can broadcast up to 50,000+ events per second to up to six clients.

Conclusions and Recommendations

Performance and scalability are crucial concerns for enterprise applications. These benchmarks show that Voyager's performance and scalability on commodity hardware is sufficient for highly demanding systems and is far greater than Axis's capabilities.

Web Services is becoming the standard for developing distributed applications. However, performance and scalability of Web Services products may be insufficient for some of today's large-scale enterprise applications.

We recommend that Web Services be utilized for the following situations:

- Across system boundaries to provide loosely coupled interoperability at a coarse-grained level
- To expose services over the Internet
- To interface with or provide an interface to 3rd-party applications

Conversely, we recommend utilizing a high-performance protocol such as Voyager for communication in large-scale enterprise applications where performance and scalability are critical.

About the author

Thomas Wheeler is a senior software engineer at Recursion Software, Inc. He may be contacted by email at engineer@recursionsw.com.

VOYAGER®

Recursion Software, Inc.
2591 North Dallas Parkway
Suite 200
Frisco, Texas 75034
1.800.727.8674 or 972.731.8800
www.recursionsw.com


RECURSION
SOFTWARE, Inc.