

BUILDING A LOW LATENCY INFRASTRUCTURE FOR ELECTRONIC TRADING

A Sun Microsystems Perspective

Table of Contents

Building A Low Latency Infrastructure For Electronic Trading	3
Introduction.....	4
The Electronic Trading Landscape	5
Trading Systems Architecture and Latency	8
The Trading Chain and Latency Hotspots.....	24
Conclusion	27
Sun’s Low Latency Partner Eco-System.....	28
Further Reading and Resources For Low Latency.....	29

Building A Low Latency Infrastructure For Electronic Trading

A Sun Microsystems Blueprint

As the global financial markets continue to shift towards electronic trading, the speed at which trading is able to take place establishes the winners and the losers among participants in the trading chain.

Latency, wherever and however it is introduced, is the enemy of speed. Market participants are continuously driving to reduce latency in order to stay at the top of the competitive trading landscape. And in order to trade profitably the top is the only place to be.

At the same time, developments such as algorithmic trading and market fragmentation have caused transaction and market data rates to skyrocket. For example, message rates across North American exchanges now routinely exceed a million messages per second at market openings.

Coping with such peaks places considerable stress on market data distribution and trading systems and generally has a negative impact on latency. The challenge for systems designers is to build systems to deliver lowest latency, even during extreme trading conditions.

This white paper examines the hotspots within an electronic trading environment and considers routes and technologies which might contribute to reducing or even eliminating latency.

“Financial institutions facing the challenge of minimizing latency in their trading systems are turning to Sun for solutions. Sun’s high performance server and storage systems, the scalable and robust Solaris 10 operating system, real-time Java and high productivity development tools, coupled with industry leading partner products and services provide a unique offering for firms meeting the latency challenge, whilst still reducing power requirements and total cost of ownership.”

Ambreesh Khanna

Head of Financial Services
Sun Microsystems

Chapter 1 Introduction

As the global financial markets continue to shift towards electronic trading – where computerized execution venues match orders received from computerized algorithmic trading engines at market participants – the speed at which trading is able to take place establishes the winners and the losers among participants in the trading chain.

A Simplified Electronic Trading Chain

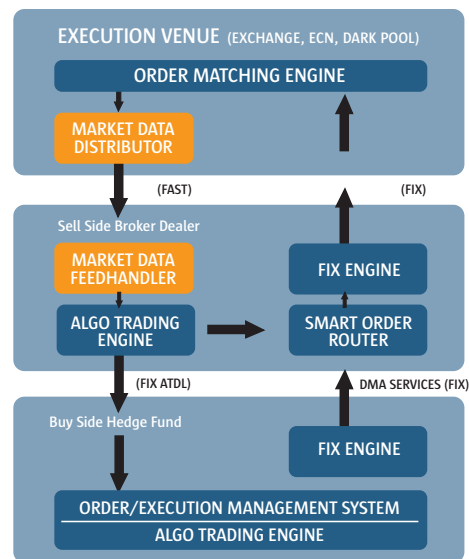


Figure 1. A Simplified Electronic Trading Chain

Execution venues – including traditional exchanges, Electronic Communication Networks (ECNs) and Dark Pools – are increasingly competing for order flow by how fast they can match orders and disseminate trade confirmations. Broker/Dealers compete on how fast their algorithms can determine trading opportunities and deliver orders to execution venues, and also on how efficiently they can service their institutional clients who are demanding the fastest execution of their orders.

Latency in all parts of the trading chain is the enemy of speed. Minimizing latency, wherever and however it might occur, is the focus of trading and IT executives throughout the financial trading markets. Time – measured in milliseconds and even microseconds – means money.

This paper outlines the electronic trading landscape, identifies specific applications that underpin the trading chain, and discusses latency issues and approaches to minimize it along the complete trading chain by leveraging Sun Microsystems hardware, software, and services offerings and leveraging Sun’s partner ecosystem.

Chapter 2

The Electronic Trading Landscape

Electronic trading – computer to computer transactions encompassing execution venues, sell side broker/dealers and their buy side institutional clients – continues its rapid rise, expanding across asset classes and geographies.

In US equities alone, some 53% of trading in 2010 will be driven by computer-based algorithms, up from 33% in 2006, according to research firm Aite Group. Already, other markets, including options on equities, foreign exchange, and fixed income trading, are increasing their use of algorithms. Meanwhile, algorithmic trading in Europe and in Asia/Pacific is also on the rise.

Algorithmic trading has evolved from leveraging relatively simple strategies, such as VWAP (Volume Weighted Average Price) and Arrival Price, to complex ones designed to underpin specific trading goals. Some algorithms are focused on the generation of orders to generate returns (alpha and beta) while others are designed solely to execute orders with the minimum market impact.

While different trading markets and algorithmic approaches are more or less sensitive to the speed of execution, it remains an important factor across all electronic trading markets. The diagram below illustrates how latencies typically measured in milliseconds (but as low as microseconds in some cases) impact trading strategies.

Typical Latencies in Electronic Trading

100s of msec	10s of msec	msec	µsec
Low Frequency Alpha Trading	FX and FI Market Making	Derivative Pricing	Latency Arbitrage
	Prime Brokerage Services	Equities DMA Services	
		High Frequency Beta Trading	

Figure 2. Typical Latencies in Electronic Trading

Source: Citihub Report 'Latency Remediation in eTrading Platforms'

The development of the more complex algorithms remains mainly a function of major sell side firms (and specialist consulting groups), but those firms are increasingly offering them to second-tier players, which 'white label' them as their own.

Algorithms are also being adopted by buy side customers, both traditional and hedge funds. For the sell side, offering algorithmic services to the buy side is a promotional tool (complete with slick names, such as Summit Runner, Covert and Prowler) that they hope will be reciprocated with order flow.

But buy side firms are increasingly consuming algorithms from several sell side firms, delivered and integrated via trade order management systems from the likes of Fidessa LatentZero and SunGard Brass, or perhaps via the FIX Algorithmic Trading Definition Language. These buy side firms will typically route tranches of small-size orders to execution venues via multiple sell side routes, using DMA (Direct Market Access) networks, to achieve the lowest visibility of their trading strategies.

Alongside the increasing use of algorithms, and driven by regulations such as Reg NMS in the USA and MiFID in Europe, the number of execution venues is also proliferating. And competition between them for order flow has become intense. As well as competition between traditional venues, new entrants include ECNs and Dark Pools.

One key area in which execution venues compete is on the speed of their trading services, including how fast they can match orders and issue confirmations of trade execution. Round trip latencies (order input to trade confirmation) of less than 10 milliseconds are common for customers geographically close to the venue.

In addition, sell side firms compete with the one another on how fast they can route orders – usually formatted to FIX protocol standards – to venues via their DMA services.

For both execution venues and sell side firms, offering not only low latency services but also consistency of latency – throughout the entire trading session – is important, especially for algorithms that build latency into their computations. High availability is also a requirement because the ultimate form of latency is caused by system failure, and the impact could be that the firm is out of the market.

In addition to minimizing latency, trading systems also have to be engineered to cope with an explosion in market data message rates, the result of a decrease in order sizes (driven by algorithmic trading), increase in number of trading venues, decimal pricing, and multiple listing of options. As an example, aggregate trade and order notifications across North American equity and options exchanges has seen recent peaks of more than 1.5 million messages per second (according to the monitoring service marketdatapeaks.com).

Market data vendors in the real-time space, such as Thomson Reuters and Interactive Data Corp. are addressing these low latency and message volume challenges by not only substantially upgrading the capacity of their own consolidated data feeds, but also by offering direct to execution venue feeds, which are then consolidated at a customer's premises as part of a managed service.

It is worth noting, however, that even for managed direct feed services the complexity of consuming direct feeds is significant since processes such as data normalization and enrichment need to be taken care of by the customer. And since such processing takes time, simply comparing the latencies of consolidated versus direct feeds is not a particularly useful measurement.

The major execution venues and sell- and buy-side firms are investing considerable sums in IT to minimize latency and maximize throughput within their trading systems, in what has become known as a “Low Latency Arms Race.”

For developers, highly optimized proprietary code is the norm, being deployed on high performance multi-core and multi-processor hardware, leveraging in-memory databases. In some areas, specialist co-processors such as Field Processor Gate Arrays (FPGAs), Graphics Processing Units (GPUs), and similar technologies have been deployed for applications such as market data handling and running derivatives pricing calculations.

However, especially for algorithmic trading, the rapid introduction of new versions of algorithms and new trading strategies creates challenges for developers. These are best met using mainstream technologies that are supported via an array of development tools including compilers, testing and debugging harnesses, and profiling offerings.

Execution venue to sell side connectivity historically has been via very high capacity telecommunications circuits. Increasingly, however, execution venues and third parties are offering co-location services, allowing customers to locate their algorithmic trading engines physically close to the venue’s market data feeds and matching engines, minimizing the distance between them and consequently reducing latency even further.

Chapter 3

Trading Systems Architecture and Latency

Latency is a factor at each and every point in the electronic trading chain, to a greater or lesser extent. The challenge for the architects and developers of trading systems is to determine where most latency exists, since it follows that making improvements in these areas will result in the most significant savings.

But no two trading operations are the same. The latency profile of a major sell side firm engaged in trading in global markets and across multiple asset classes is going to be very different from a specialist hedge fund focused on trading a subset of securities on one or two exchanges.

For each firm, very different trading applications and IT infrastructures would be deployed, with different requirements in terms of market data, telecommunications, inter-firm networking, datacenter hardware, storage, and operating software. And, of course, very different applications software and support requirements.

Latency is also not a constant but is impacted by many factors. Transaction and message volume fluctuates based on regular events, such as market openings, and also on market events, such as breaking news or the execution of an algorithm. Spikes in message rates – and a corresponding increase in input/output interrupts – tend to impact latency and cause it to vary from normal levels; a phenomenon known as jitter.

Minimizing jitter, which can also be caused by poor network design, network capacity limitations, hardware and operating system configuration, and the design of applications, is becoming a major focus since its impact is generally felt most when opportunities present themselves to exploit fast moving markets for profit.

To drill down on latency hot spots, it is useful to consider a trading systems architecture as consisting of three layers: connectivity and networking; IT infrastructure; and the trading applications themselves.

- **The Connectivity and Network Layer** – connectivity to trading venues including telecoms, propagation, serialization, IP protocols, network equipment (switches, routers and firewalls).
- **The IT Infrastructure Layer** – messaging, market data distribution, the FIX protocol, storage/persistence, grid computing, operating system, hardware (CPU, memory, I/O).
- **The Trading Applications Layer** – consisting of both packaged applications, such as order routing and event processing packages, and proprietary applications including derivatives pricing and portfolio evaluation functions and trading algorithms.

Trading Systems Architecture Layers

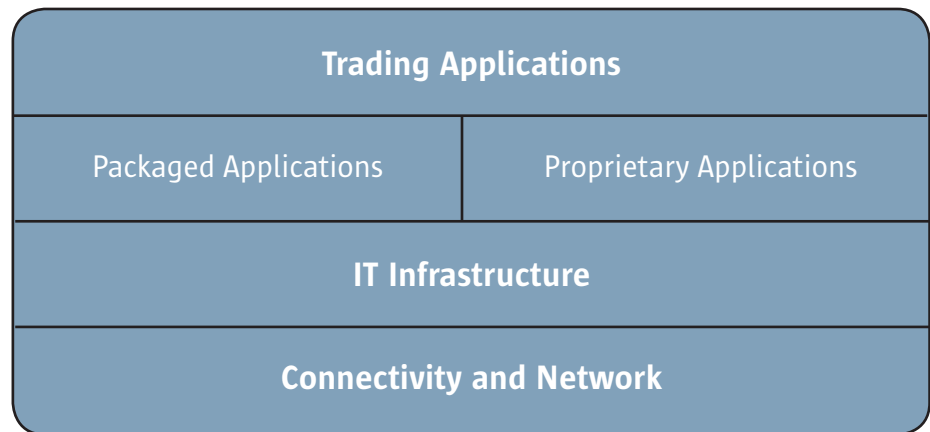


Figure 3. Trading Systems Architecture Layers

Notably, a recent industry report* examined levels of latency across the trading chain for a number of different trading markets, and concluded that common across all of them, application layer contributes by far the most latency to the end-to-end total – between 50% and 70%.

Connectivity and Networking

Inherent in network latency is that which exists as a result of the distance between the trading firm and the execution venue, and the type of telecommunications media. For optical fiber circuits, data travels at about 100,000 miles per second. This equates to latency of about 10 microseconds over a distance of one mile, and a millisecond when 100 miles separate sender and receiver.

Increasingly, execution venues – often in partnership with hosting services, such as BT Radianz, Fixnetix, and SAVVIS – are offering collocation or “proximity” services, allowing trading firms to locate their electronic trading systems physically close to the exchange order matching systems. Such initiatives are fundamental to the business models of execution venues which are seeking to maximize order flow and hence liquidity. They exacerbate the complexity to those firms looking to execute strategies across many venues, since latency will vary from one to another.

But taking advantage of these proximity services comes at a price. Datacenter space close to execution venues is finite and demand for it is increasing. So supply and demand factors into base pricing inputs, which include physical floor space taken and power consumed to run and cool the hardware.

Sun has a range of offerings – including rack mount and blade servers, storage, operating systems, virtualization, and professional services – designed to maximize performance and minimize power and cooling requirements, and, hence reduce total cost of ownership (TCO).

Compared to similarly priced products from competitors (across a range of price points), with equal computer performance, Sun's rack mount server products require half of the rack space (e.g. 2U of space versus 4U for competitors). Thus, rack for rack, Sun servers deliver twice the compute power of competitors. Moreover, Sun's servers draw as much as 45% less power and dissipate more than 20% less heat compared to competitive offerings. Combined with rack space savings, TCO savings for typical installations might be measured in millions of dollars – see specific products at sun.com for more detailed information.

Blade servers are an alternative where space is at a premium. A single Sun blade chassis allows dense packing – currently up to 48 per rack – of compute blades, providing a compact hardware platform capable of running a full suite of trading system applications under different operating systems. Integrated storage and dedicated I/O blades can also be deployed in the same rack as compute blades, to boost performance and avoid bottlenecks that impact blade products from other vendors.

In particular, Sun offers the 'Sun Compute Cluster for High Performance Co-Lo Trading', a pre-configured, ready-to-deploy solution incorporating server, storage, networking, operating system, systems management software, and diagnostic tools. Specific features include:

- Fast deployment - Sun programs combine Sun and approved third-party products into complete systems through factory integration, ready for deployment upon arrival. Ease of management via preloaded tools for provisioning and administration.
- Flexibility to mix and match server (AMD™/Intel®/SPARC®) modules based on application workload with any operating system of choice all running on and managed in the same rack.
- Scalable compute power — scaling from 1 to 8 racks and 30 to 240 nodes (blades) or 32-256 nodes with 1 RU rack mount servers.
- Ability to choose between multiple configurations of storage (network attached or direct attached) based on the application workload and the latency requirements. Block storage can be provided to each node for low latency applications such as pricing and algorithmic trading. These applications require large numbers of I/Os per second (IOPs) combined with low I/O response times. File-based NAS storage is provided as a shared resource for reference data and tick data repositories.
- Solaris™ 10 as the base operating system, including Containers and DTrace for low latency virtualization/server consolidation and for observability, analysis, and performance tuning of home-grown and off-the-shelf trading applications.

Sun's Container virtualization technology adds further to the performance/cost equation, allowing server utilization to be maximized by running multiple virtual server workloads on the same physical server. Also, unlike other virtualization

approaches, Containers do not introduce a performance penalty. In fact, input/output between Containers offers superior performance over network-based communications – exploiting the low-latency high-bandwidth interconnects within multi-core, multiprocessor systems – so distributed trading application performance is in fact improved.

Since trading systems typically comprise many – tens or even hundreds – of servers acting as a unified distributed processing environment, so networking and network latency are issues to be considered.

The common use of IP protocols across Gigabit or 10-Gigabit Ethernet can increase latency due to the processing required to support reliable data transfer. Sun's Solaris operating system allows tuning and optimization of IP protocol handling, for both UDP and TCP, such as buffer sizes and highwatermarks. In addition, Solaris allows operating system functions such as interrupts to be segregated from the applications that depend on them, thus minimising overheads from context switching. All this serves to reduce latency without compromising reliability; Solaris is able to sustain high network throughput and low latency without dropping packets – an important consideration for applications such as market data and trading and not always true of other operating systems.

InfiniBand networks, which can use TCP/IP but may also employ proprietary RDMA protocols, allow data to be transported between servers without interrupting the CPUs within them. Unlike Ethernet, this provides for deterministic latency between nodes, and is thus a popular choice for high performance interconnects between servers located within the same datacenter. As well as offering its own InfiniBand range of products, Sun partners with the major InfiniBand vendors - Voltaire, Mellanox Technologies, and Qlogic - and provides support for Infiniband in Solaris.

IT Infrastructure

IT Infrastructure covers everything that is not related to networking or the trading applications. This includes the hardware platform (servers and storage), operating system, virtualization, and messaging middleware layer.

Sun's Server Architectures

Sun's server products span two separate architectures – x64 and SPARC, each providing specific functionality and performance characteristics, and each applicable to the workload profiles of different financial trading applications. Within a single end-to-end trading system both x64 and SPARC-based systems would typically be deployed. Communication and collaboration between the architectures is seamless, since both are optimized to run the Solaris operating system.

Server products are available in both rack mount and blade formats. As previously noted, Sun servers minimize TCO, delivering twice the compute power per rack compared to other vendor offerings, while drawing 45% less power and producing 20% less heat. Blade servers allow for high compute density packing – up to 48 multi-core x64 or SPARC blades per rack – combined with storage and I/O blades to maximize performance.

Servers based on x64 architecture – from Intel and AMD – are best suited to the high throughput, single threaded applications that benefit most from the fast clock speeds and CPU cache memory that are found on x64-based systems. Market data feed handlers, messaging middleware and FIX engines often fall into this profile as do execution-oriented algorithms, exchange matching engines, and smart order routers.

As an example, a recent benchmark conducted for the Thomson Reuters Market Data System, on Intel-based servers running Solaris and connected with 1GB Ethernet, achieved one millisecond end-to-end latency at a throughput rate of 500,000 messages per second. This performance would be further enhanced by using either 10GB Ethernet or Infiniband.

Sun and Intel Strategic Partnership

In January 2007, Sun and Intel announced a wide-ranging design, engineering and marketing partnership designed to position Solaris as the premier enterprise operating system for Intel's x64 hardware platforms, both current and future.

Engineering teams from the two companies collaborate to optimize Solaris for the new Intel Core Microarchitecture and for the Intel® Xeon® processor; with compelling results. The goal ensures that applications running on Solaris/Xeon are faster, more scalable, and draw less power than those running on other platforms or operating systems.

“Intel is focused on engineering architectures which fully optimize the features of our processors to achieve lowest possible latency in trading operations. This involves close collaboration with Sun around both hardware design and the unique features of Solaris 10 for the benefit of our joint customers,” said Nigel Woodward, Global Director, Financial Services, for Intel. “Sun has invested in the Intel fasterLAB program to facilitate this joint engineering activity.”

Some key features of the collaboration include:

- Increased performance as Solaris leverages Intel multi-core processor capabilities. Solaris can leverage up to 256 processing cores, each running two threads for a total of 512 threads per server.

- Support for Turbo mode, which converts any available power headroom into higher frequencies. In those situations where Solaris determines that maximum processing power is required, the processor increases the frequency in the active core when conditions such as load, power consumption and temperature permit it.
- Leverage of integrated memory interfaces for faster memory input/output. In addition, very large physical memory spaces are possible with support of the Xeon's 44 bit memory addressing.
- Use of specialist instructions, such as string handling and data compression, highly relevant to running market data handling applications.
- Optimized power efficiency and utilization by enabling Solaris to take advantage of the new Intel Core Microarchitecture performance-enhanced dynamic power management capabilities, including processor clock speed step-down.
- Improved reliability by incorporating new Intel Core Microarchitecture features into the Solaris Fault Management Architecture (FMA).
- Cost-effective virtualization by enhancing the Solaris OS to take advantage of the latest Intel Virtualization Technology features, which minimize overhead for features such as Solaris Containers.

As Intel continues to roll out its “tick/tock” development program (every second year introducing a new manufacturing process, and every other year a new microprocessor architecture), the Sun/Intel partnership ensures that Solaris will always be first in line to take advantages of enhancements to the hardware platform.

Meanwhile, the SPARC architecture is ideal for the complex high throughput applications that lend themselves to parallelization. Sun has aggressively pursued chip-level parallelism as well as the software techniques to exploit this. All of Sun's microprocessors are multi-core; UltraSPARC IV provided two CPU cores on a chip in 2003, while the latest high-end systems – the SPARC Enterprise M-Series – provide four dual-thread cores on each chip in systems up to 64-sockets.

The slower clock speed saves power but overall throughput exceeds its contemporary competitors. At the same time, the issue of memory latency has been eliminated by switching between threads while others wait for memory. This approach scales well with Moore's law, as both replication of CPUs and the memory bandwidth to feed them scale together more readily with increasing transistor density than with simple increases in clock speed.

Future SPARC developments (such as the 16 core “Rock” chip) add even more innovation with parallel hardware that can accelerate performance of individual threads.

Complex alpha/beta algorithms, derivative pricing, portfolio and risk analytics applications can benefit from and scale well on parallel compute architectures. Transactional applications also lend themselves to multi-threading SPARC environments.

Low Latency Storage

In trading systems design, the need to store and retrieve data rapidly as part of the transactional process – also known as persistence – and for auditing and later analysis has historically been a challenge due to the impact on latency and the high costs of reducing it.

Persistence is a requirement in such cases as FIX and other guaranteed delivery communications, where the ‘state’ of each transaction thread needs to be stored in order to facilitate recovery from errors.

Market data feed handlers often need to store messages in sequential order as it arrives – so called tick data – for later replay to analysis applications or for best execution regulatory needs. They also often maintain last value and high/low caches.

Traditional storage, such as electro-mechanical hard drives, introduces latencies in the order of milliseconds and so is not an option for low latency applications. Random Access Memory (RAM), while being fast – in the order tens to hundred of nanoseconds – is extremely expensive and its capacity is limited. Hence, its impact on low latency applications has been limited.

Sun’s Solid State Disk Array introduces the most dramatic change to the storage hierarchy for decades, providing non-volatile multi-terabyte storage capacity with access times two-to-three orders of magnitude faster than disk-based storage, typically around ~0.1 milliseconds or less. Big improvements to latency can be achieved without the need to alter the application code since the Solaris ZFS file system inherently leverages solid state disks, either to hold entire file systems or as a cache to accelerate reads and writes to hard drives. ZFS also lowers storage cost by reducing the number of spindles, footprint, and power consumption versus traditional storage.

In conjunction with ZFS, Sun has introduced a range of integrated storage servers, which tightly couple compute and storage resources within the same unit. The first generation of these devices combine dual and quad-core CPUs, with DRAM and disk

storage (up to 48 terabytes) in a closely integrated 4U unit. In addition, the Sun 7000 Unified Storage System is a new type of NAS appliance that offers more performance, lower cost, and easier management than competing products.

The Solaris Operating System

Building on a 20 year design heritage, Sun's Solaris is a robust, scalable operating system able to run large numbers of processes and threads across multiple CPUs, offering high input/output performance.

Available for free download, the fully supported enterprise version of Solaris is augmented by the open source OpenSolaris, involving a global community that is working as a group on new functionality and enhancements that are destined for the enterprise edition. This enterprise/open source approach allows developers to get access to, and experience with, new features.

Sun guarantees Solaris to be backwards compatible, a huge boost for application developers and independent software vendors, since maintenance is usually the largest expense associated with software. With this backwards compatibility guarantee, software vendors know that software built for Solaris version N will run correctly on Solaris version N+1 and subsequent versions without modification. This allows for the higher performance of subsequent releases to be leveraged immediately without the need for costly and time consuming regression testing, and speeds deployment of applications as new versions of Solaris are introduced.

Solaris was designed from scratch to support real-time applications in a multi-core multi-processor environment, with both high performance and determinism to maximize throughput and minimize latency and jitter. It is the years of experience of operating in such stressful environments which has resulted in Solaris being such an accomplished enterprise operating system, and so advanced when compared to relatively new arrivals such as Linux.

Added resilience - fault management and predictive self-healing pre-empt system failures allow Solaris to identify and isolate faulty hardware before it causes unplanned downtime. This innovative feature is the result of Sun's decades of experience with large enterprise systems and a close relationship with Intel and AMD to understand the behaviour of critical components, and to create the "telemetry" mechanisms to monitor these in software, and the self-healing mechanisms to protect against faults.

Solaris is fully featured and API compatible across both its x64 and SPARC variants – allowing it to run as the basis for all applications in a trading system, and reducing systems management and TCO, which becomes significant in multi-server distributed systems. Solaris is available and supported on servers from IBM, Dell and others, making it an obvious enterprise operating system choice for heterogeneous environments.

Also, the operating system is optimized end-to-end for both the x64 and SPARC platforms, from the kernel and libraries to the Java™ Virtual Machine (JVM) and Sun developer tools, allowing customers to make their hardware platform choice based solely on application requirements.

Specific Solaris performance highlights include:

- Containers, offering an innovative approach to virtualization. Maximizes server utilization and improves inter-process communication performance without the overhead associated with hypervisor approaches.
- The Zetabyte filesystem (ZFS) providing ground-breaking performance and near-zero administration.
- TCP/IP networking stacks that boost application performance by around 50% compared to alternative Unix or Linux offerings.
- Processor binding to ensure threads run on the same processor (or core within a multi-core environment) to improve performance and maximize system utilization.
- Multiple levels of scheduling priority, allowing low latency applications to run without interruption even from the operating system kernel, maximizing performance in a deterministic manner, and minimizing jitter.
- Memory locking and placement optimization, which boost application performance and improves determinism.

Many performance parameters within Solaris are tunable, dependent on specific application workloads, in order to optimize performance. Specific examples where Solaris tuning has delivered significant performance improvements include:

- A major broker/dealer, where roundtrip order execution time for customers was reduced from 90 milliseconds to less than 10 milliseconds.
- A US regional exchange, where market data throughput was increased from 250,000 messages per second to 900,000 messages per second.
- A US options exchange, where network latency was cut by 50%.

Solaris Containers Virtualization

Unique to Solaris – and so free - is its Containers functionality. Containers are a unique form of virtualization technology that allows multiple application stacks to be run on top of a single Solaris instance, without the performance overheads associated with hypervisor approaches.

Solaris Containers allow individual instances of Solaris to be “installed” and “booted” (each instance called a non-global Container) within a single global instance of the Solaris operating system. Each of these Containers appear as individual Solaris instances, each with their own identity, user namespace, IPC namespace and network namespace.

Containers are security and fault contained, thus one Container does not have the ability to view the data or resources associated with another Container. Faults inside one non-global container cannot propagate into any other Container, improving system stability.

Resources may be dedicated to Containers. These resources include processors and processor cores, memory and network bandwidth. Each Container also has the ability to run with a different default scheduling class, thus affecting runtime behavior for only the applications running within that Container.

File systems can be dedicated to Containers, and can also be configured as shared resources between containers. Physical devices, such as tape devices and raw disk devices, can be dedicated to Containers.

Communication between Containers employs a technology called TCP Fusion, which emulates standard network communication between processes, so applications do not require modification to run in a Container environment. TCP Fusion improves application performance by eliminating network latency between physical servers. Because of this performance boost, Thomson Reuters has endorsed Containers as the only virtualization approach supported by the Reuters Market Data System (RMDS).

Benchmarks run on a “real life” RMDS deployment scenario that can handle one million messages per second involved consolidating eight physical servers into just one, provided more than 2.5 times better performance per Watt and two times better performance per rack unit.

Containers can also be used to consolidate multiple instances of similar applications onto a single server to reduce hardware footprint and associated space, power and cooling costs - reducing up to 50% of space and up to 60% reduction in power consumption, according to recent benchmarks.

As well as reducing the hardware requirements of RMDS, Sun has consolidated large number of Options Price Reporting Authority (OPRA) feeds on Solaris using Containers, with near linear scale.

Messaging Middleware

Messaging middleware is the essential communications glue that binds trading applications together, building on the basic functionality of Ethernet, InfiniBand and IP to provide reliable delivery, guaranteed sequencing and transaction semantics.

Sun works closely with leading vendors of messaging middleware, all of which are addressing the need for low latency and high throughput, to ensure that their offerings run best on Sun hardware and the Solaris platform.

Thomson Reuters Market Data System (RMDS) is the most widely deployed middleware in the financial markets and when coupled with the Reuters Datafeed Direct ticker plant offers a complete low latency solution for connectivity of trading applications to exchanges and other liquidity venues. Benchmarks of RMDS running on Solaris consistently set new world performance records.

Sun also partners with other messaging middleware vendors including Tibco Software and 29West to ensure that their offerings perform best on the Sun hardware and operating system combination. For example, 29West middleware is being deployed around the globe in the most demanding applications where performance and reliability are critical. 29West's LBM and UME products are optimized for Solaris, with many customers deploying on Sun infrastructure for great performance and proven reliability.

Trading Applications

Trading firms face extreme business challenges when building trading services. Firstly, there is a time-to-market element in supporting new execution venues (which might never attract significant liquidity) and offering low cost access to them. Second, transactions need to be executed faster than the competition, or fail to be filled.

For execution venues, the challenges are similar. Providing 100% uptime, combined with the lowest transaction roundtrip is mandatory to attract and keep order flow. As new trading venues are created, and as standard interfaces including FIX are adopted, traditional ones are becoming acutely aware that liquidity built up over years can be lost in weeks, days or even hours.

Electronic trading systems – wherever they be deployed – typically combine vendor packages with proprietary code working in unison. The use of packages allows systems to be built rapidly, and such packages might deliver a complete point solution, such as Smart Order Routing applications, or they might be a building block, as would be the case for complex event processing engines and market data feed handlers.

Proprietary code will likely be written for functionality such as order matching and algorithmic trading, where firms are seeking to create a specific advantage that will mean the difference between winning business or losing it. Whether code be a package or proprietary, developers face challenges in making it as efficient as possible and yet easy to integrate with other modules. Ease of maintenance is also an issue, since maintenance costs are often the largest cost associated with developing software.

Developer Tools

Sun provides a number of tools that allow developers to design, code, debug and tune applications to minimize the latency within them.

These tools include the Distributed Make – DMake – utility, which allows a developer to concurrently distribute the process of building large projects consisting of many programs, over a number of workstations and, in the case of multiprocessor systems, over multiple CPUs.

Sun compilers for C, C++, Fortran and Java produce code parallel optimized for multi-core environments. Additionally, Java code can be optimized to make use of Real-Time Java functionality, which improves runtime performance and reduces jitter.

Developers and systems designers can then use Dynamic Tracing (DTrace) functionality to profile the performance of running applications, whether package or proprietary, allowing real life performance to be determined and improved (see box - Fine Tuning SunGard BRASS With DTrace)

Parallelism with Multi-Core and Grid Cluster Architectures

While several applications that form parts of the trading chain are better suited to single core, single thread architectures, applications such as algorithmic trading engines and order matching have the potential to exploit parallelism – but this requires the right tools.

Within a single system, parallelism can be exploited for multiple processor cores on a single CPU die and for multiple CPUs in the system. OpenMP is a programming model that supports this – within standard languages such as C, C++ and FORTRAN – simply by adding directives in the form of pragmas or comments. Parallelism among multiple systems requires an API for communication and synchronization; in this case the MPI (Message Passing Interface) standard dominates. With years of experience in parallelism for High Performance Computing, Sun has played an important role in developing and implementing these and related open standards.

Both of these approaches – spanning the full gamut of parallelism – are supported by Sun Studio, Sun's development environment including compilers, debuggers and performance diagnosis tools for C, C++, and Fortran, running on both x64 and SPARC architectures and based on the NetBeans IDE (with full command-line control for those who prefer a more traditional interface).

Studio is developed in close collaboration with CPU developers and optimization experts at Intel and AMD, as well as the Sun and Fujitsu SPARC teams; this ensures that Studio leads the pack in performance and features.

The latest version of Sun Studio Express includes:

- Aggressive compiler optimizations for the newest multi-core architectures including Intel's Nehalem and AMD's Shanghai, as well as Sun's highly-parallel UltraSPARC T2 and SPARC 64 VII.
- Thread analysis tools to quickly identify complex multi-threading issues.
- Compiler auto-parallelization of single-threaded code.
- OpenMP version 3.0 support, including support for Tasks – a programming construct that takes OpenMP beyond loop-level parallelism to support more complex data structures such as recursive trees and linked lists.
- MPI2 support via Sun's implementation of OpenMPI version 1.2 support in the ClusterTools 8.0 library, supporting up to 1024 nodes and 4096 processes.

As well as compiler support, debugging and performance analysis, tools allow developers to:

- Deep dive the performance of code, algorithm changes and hardware system counters.
- Correlate source code to actual machine execution.
- Profile Dataspaces on UltraSPARC systems, to determine performance hits related to application memory references.
- Fix code 'on-the-fly' in a running program without having to recompile the entire application.
- Profile parallel applications that use OpenMP or MPI, as well as single-threaded applications.
- Attach a debugger to a running application using dbxtool.

Studio supports several flavours of Linux, in addition to Solaris 10 and OpenSolaris. Studio incorporates Intel's Threading Building Blocks (TBB) library for C++, which allows developers to take advantage of multi-core processor performance without needing to be a threading expert. TBB abstracts platform details and threading mechanisms and makes parallelism straightforward at the application level, for performance and scalability.

Trading Application Architecture – GigaSpaces XAP

GigaSpaces' eXtreme Application Platform is an application server built to scale that provides you a way to see your entire set of computers as one single, simple runtime environment in which both your processes and your data can reside, enabling predictable improvement of application performance while you increase the volume of data, transactions, and number of users, with no real need to rearchitect. XAP is an alternative to traditional application servers, providing predictable, ultra low latency and the high throughput performance required by applications such as order management and routing, trade matching and execution, market data processing, and real-time analytics.

In a recent benchmark for GigaSpaces XAP running on a Sun x64 server, a typical risk application ran at a throughput of 1.7 million reads/second and 1.1 million writes/second and demonstrated latency of 1ms under a transactional, highly concurrent workload and full replication. The same platform supported an online application workload of over 16,000 page generations per second with 3 web servers, 100 concurrent users and a latency of under 0.6 ms over LAN.

In another recent benchmark on a two CPU Sun SPARC server, a GigaSpaces XAP enabled financial services application processed 1.3 million transactions per second against 60 concurrent clients.

XAP employs a three-pronged approach to achieve unparalleled levels of transactional low latency:

- Faster access to data via an in-memory data grid.
- Proximity of tiers and elimination of bottlenecks via co-location of data and messaging close to the related business logic.
- Faster and more scalable thanks to a dynamic, self managed parallel processing framework, with seamless failover and recovery.

GigaSpaces has optimized a “Low Latency Stack” with Sun and Intel, the networking vendors, and other middleware and messaging players. This includes SPARC and x64 multi-core optimizations, optimizations with Gigabit Ethernet and InfiniBand, optimizations at the Sun Solaris and Sun Java Virtual Machine levels, and Sun CoolThreads, to ensure a robust platform with no weak links.

Real-Time Java

Although Java enables fast application development, the adoption of Java technology in real-time and low latency domains has traditionally been limited by the inability to predictably control the temporal execution of Java applications, due to:

- Unpredictable latencies introduced by automatic memory management (garbage collectors)
- Inadequate scheduling control
- Unpredictable synchronization delays
- Very coarse timer support
- No asynchronous event processing
- No “safe” asynchronous transfer of control

These issues have been addressed by the Sun Real-Time Java System (Java RTS), which can boost the performance of low latency applications with minimal code changes.

Supported on both SPARC and x64 systems running Solaris, Java RTS performance is measured in two ways: throughput performance for non-real-time logic, and predictability of performance – within the acceptable boundaries of latency and jitter – for hard real-time logic.

For non-real-time logic, Java RTS has been seen to perform up to 85% as fast as the equivalent non-real-time system on standard Java benchmarks. For hard real-time logic, Java RTS on reference platforms has a maximum latency of 20 microseconds and a maximum jitter of 10 microseconds.

Java RTS incorporates specific functionality to decrease latency and jitter, including:

- Real-time and no-heap threads, which cannot be interrupted by garbage collection. Garbage collection is traditionally cited as the principal drawback of deploying Java in real-time environments, so Java RTS removes that drawback completely.
- A new memory management scheme, including ‘immortal’ memory for static objects, thus avoiding garbage collection and the delays caused by it.
- Asynchronous event handling, including safe thread pre-emption and controlled responses to external events.
- Timers with nanosecond accuracy.
- Direct access to physical memory.
- DTrace hooks for application performance profiling.

By controlling garbage collection and interrupt handling, applications written in Real-Time Java are highly deterministic in nature, minimizing application-induced jitter whilst continuing to deliver the benefits of developing applications in Java.

Dynamic Tracing (DTrace)

Dynamic Tracing (DTrace) is a Solaris tracing framework designed for troubleshooting systemic problems and bottlenecks on production systems in real time. DTrace can monitor both the Solaris kernel and running applications without system reboot, application restart or recompiling.

Sun has created custom DTrace scripts for Thomson Reuters RMDS and NYSE/Euronext Advanced Trading Solutions Wombat middleware. Customers can also create scripts to monitor and tune their own applications.

For example, in a trading system, DTrace allows the tracking of a message through different applications and the message bus, FIX engines, etc. so that bottlenecks can be detected across the entire trading chain.

Fine Tuning SunGard BRASS With DTrace

SunGard BRASS is a comprehensive solution for trade order execution and management, which is available for both Sun x64 and SPARC architectures.

Sun and SunGard engineers employed an iterative approach that consisted of using DTrace to monitor and analyze an existing BRASS system end-to-end under a typical workload in order to pinpoint possible areas for improvement.

Using DTrace, several performance issues were identified, including process and memory bottlenecks, processor utilization and network packet transmission.

As a result, it was possible to tune both Solaris and BRASS so that it was able to handle four times the input message rate.

Chapter 4

The Trading Chain and Latency Hotspots

Several applications make up the entire trading chain and each performs specific functions with different latency profiles and requirements. Major applications are discussed below.

Liquidity Venue / Exchange Order Matching

Order matching systems maintain a 'book' of both sides of an order (bids and offers) across a number of securities. For each new bid or ask received, it is the task of the matching software to scan the entire book as quickly as possible to match it to alternate sides of the transaction. While the matching process calls for fairly simple logic, the requirement to handle thousands of matches per second calls for very high performance.

When a match occurs, both sides to the transaction are informed and a trade report is distributed to the marketplace at large (and archived for end of day data feed services and for regulatory purposes). Single digit millisecond latency is common for major execution venues.

Orders and confirmations are typically received and conveyed via the FIX protocol and usually communication via this protocol is performed via FIX gateways. Such gateways are usually supplied by specialist vendors, such as NYFIX, Orc Software and ULLINK.

However, the matching element of the process is usually conducted via a proprietary-coded application, which is highly optimized for speed. Consistency of matching speed (that is, low jitter) is considered as important as absolute speed, as is very high availability. Efficient connectivity between the FIX gateway and the order matching engine (as well as market data distribution covered below) is a fundamental requirement.

Thus, Sun technologies such as x64 servers (for high throughput), Solaris (high performance, low jitter, high availability), Containers (fast inter-application communication) and the Real-Time Java System (for low jitter) are appropriate for developing order matching systems.

Market Data Handling

Market data handlers – whether they be at an execution venue distributing feeds to market participants (both trading firms and commercial aggregators/redistributors) or at a trading firm to receive data – perform similar functions such as very fast input/output, data normalization and error detection and recovery.

Between an execution venue and trading firms, the historical norm has been to deploy proprietary communications protocols that are optimized for high throughput (small message headers and payloads). Some venues are beginning to use the FAST (FIX Adapted For Streaming) protocol, though not all are supporting this standard to the full.

For trading firms, market data receivers would typically maintain a local cache of recent latest prices, perhaps enriching it with hi/lo information or time series data. Such functionality is of general use to trading applications, so maintaining a single cache makes design sense.

Akin to order matching, the business logic that makes up market data feed handlers is relatively simple, though once again the need for high performance is paramount (handling hundreds of thousands of messages per second from a single datafeed). Sun x64 servers are ideal for the hardware platform (Intel chipsets being generally considered as offering optimum I/O), running the combination of Solaris Containers and Real-Time Java for the operating and application software.

Sun's specialist storage servers, especially when equipped with Flash technology, might be used for maintaining caches, or audit trail time series for regulatory purposes.

Messaging middleware links market data handling components with those performing order management and trade execution. Sun's close work with messaging middleware vendors to improve performance and scalability makes it the platform of choice for such middleware.

Algorithmic Trading Engines

Algorithms come in many forms with very different purposes. Consequently it is difficult to be specific about the functionality performed by algorithmic trading engines. However, as a generalization, all are likely to require access to different data points upon which they perform analysis which triggers one or more events. The analysis can vary in complexity from the simpler pattern recognition to the application of complex mathematical formulae.

In addition to proprietary code, some functionality for complex event processing (CEP) might be acquired from a specialist vendor and would make an ideal platform on which to build algorithms. CEP engines are specialist applications that can be programmed to look for patterns of events in incoming data streams, generating alerts when specific conditions are met. Such a condition might cause some proprietary code to be executed in order to process specific algorithm rules. CEP engines generally are designed to take advantage of multi-threading, and hence scale linearly when implemented on multi-core systems. Sun partners with several

CEP vendors including Aleri, Progress/Apama, Corel8, and StreamBase Systems, and has worked with them to optimize their offerings for Sun environments. A recent benchmark running Aleri's CEP offering on a Sun Intel-based server running Solaris resulted in a data update rate of 300,000 updates per second, simulating a cross-venue order book liquidity aggregation function that might form part of an execution algorithm.

Depending on the complexity of the algorithm, Sun x64 or SPARC servers might be deployed. In either case, Solaris, Containers, and the Real-Time Java System are suitable technologies on which to base algorithmic trading, whether or not it includes vendor CEP offerings.

As noted earlier, collocation of algorithmic trading engines with execution venue matching systems can reduce latency caused by physical distance. Since collocation services usually charge by physical floor space used and power consumed, Sun server offerings – whether rack mount or blade – are an attractive option because of the high density of compute power they provide. Solaris and Container technologies can maximize the application load running on servers, while also increasing performance.

Smart Order Routing and DMA

Within a Sell Side trading firm, Smart Order Routers and DMA infrastructure is used to route orders to execution venues, either on behalf of the firm's own proprietary algorithmic trading function, or (in the case of DMA) from the firm's buy side customers.

Smart Order Routing systems ensure that orders are executed at the best price, which must be determined across a number of execution venues including dark pools. As such, they complement execution algorithms that determine how best to feed large orders into the market.

DMA provides buy-side firms running their own algorithms or using those supplied by sell-side trading partners with a fast route to the execution venues, while maintaining control over how order execution is handled.

SOR and DMA applications often fall into the high performance/low complexity area best suited to leverage x64 Servers. Once again, Solaris and Containers (perhaps allowing a SOR application and a FIX gateway to run on the same server) would form the operating software stack. Proprietary SOR and DMA applications might be written to leverage the Real-Time Java System. Alternatively, packages such as Fidessa LatentZero and SunGard BRASS – both optimized to run in Sun environments – might be considered.

Chapter 5

Conclusion

Financial markets firms are continually seeking to reduce latency and increase scalability in their trading infrastructures to gain competitive advantage, while understanding that TCO issues such as power and cooling are vital ingredients to trading profitably.

Sun Microsystems has had a focus on the trading markets since being founded in 1982, and Sun technology has been deployed globally at the heart of their trading operations by thousands of the world's leading financial institutions.

Sun partners with its customers to help them leverage a comprehensive offering, including servers, storage, networking and interconnects, the Solaris operating system, real-time Java, plus development, testing and optimization tools.

In addition to its hardware and software products, Sun brings its massive experience in designing robust, scalable and high performance systems, helping trading firms tune and configure their proprietary and third party application for fullest performance. Moreover, Sun's experience in the packaging of systems helps customers reduce both purchase and ownership costs, especially in the areas of space, power and cooling.

Sun works with partners to deliver fully optimized and supported package offerings, to speed time to market in the implementation of trading platform infrastructure.

This combination of products, service offerings and partner ecosystem makes Sun Microsystems a natural choice of partner when building the most demanding trading infrastructure to exploit the future global financial markets.

Chapter 6

Sun's Low Latency Partner Eco-System

Sun works with third party software, hardware and services partners to optimize their offerings for a Sun environment and provide complete solutions to customers.

Latency Contributor	Software Partners	Hardware Partners	Services Partners
Network Layer			Trading Applications Developer Workshop Hosting Partners Fixnetix
Middleware Layer	Reuters (RMDS) NYSE Euronext ATS (Wombat) 29 West Tibco Software	Voltaire InfiniBand Cisco 10GE Tervela Message Switch Solace Systems	Paremus CSC SunGard Hosting Reuters Hosting Fixnetix
Application Layer Decision Support Applications	Aleri Coral8 Streambase Progress/Apama Platform Datasynapse Gigaspace Paremus (Infiniflow) Greenplum SAS Numerix	nVidia GPUs FPGA Vendors	
Application Layer Order Management & Execution	Fidessa SunGard BRASS GL Trade ION Trading ULLink		

Chapter 7

Further Reading and Resources For Low Latency

Sun white papers and benchmarks

LOW LATENCY: Eliminating Application Jitter with Solaris

http://www.sun.com/solutions/documents/white-papers/fn_lowlatency_solaris.pdf

The Sun Java Real-Time System Meets Wall Street

<http://developers.sun.com/learning/javaoneonline/2007/pdf/TS-1205.pdf>

Low Latency 5 Nine's Trading: Sun's Value Proposition for Market Data

http://www.sun.com/solutions/documents/pdf/fn_mdata.pdf

Thomson Reuters Market Data System - RMDS 6.0 - Benchmark

http://www.sun.com/third-party/global/thomsonreuters/collateral/RMDS_June2008_4150_perf_x4150_s10u4_3_1.pdf

Thomson Reuters RMDS Server Consolidation with Solaris 10 Containers

http://www.sun.com/software/whitepapers/solaris10/rmds_containers.pdf

Thomson Reuters RMDS with/without Solaris 10 Containers

http://www.sun.com/third-party/global/reuters/collateral/T5120_sparc_s10u4-v1.0.pdf

GigaSpaces XAP Benchmark

<http://www.sun.com/aboutsun/pr/2008-09/sunflash.20080922.2.xml>

SunGard BRASS - Speeding transaction throughput using powerful Solaris Operating System tools and utilities

http://www.sun.com/solutions/documents/success-stories/fn_sungardbrass.pdf

Other resources

* *Electronic Trading – The Latency Challenge* from Net2S

<http://www.low-latency.com> – Low latency-related news from A-Team Group

<http://www.marketdatapeaks.com> – A Web site that provides a real-time update on market data rates

<http://www.stacresearch.com/sun> - Sun-related benchmarks from the Securities Technology Analysis Center



Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 USA Phone 1-650-960-1300 or 1-800-555-9SUN (9786) Web sun.com

© 2009 Sun Microsystems, Inc. All rights reserved. Sun, Sun Microsystems, the Sun logo, Solaris, and NetBeans are trademarks or registered trademarks of Sun Microsystems, Inc. or its subsidiaries in the United States and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the US and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc. AMD, Opteron, the AMD logo, the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. Intel(R) Xeon(R) is a trademark or registered trademark of Intel Corporation or its subsidiaries in the United States and other countries. Information subject to change without notice.