

High-Performance Business Computing with Hiperware

No representations or guarantees are made or implied. Plans and projections are subject to change. All stated amounts are approximate and estimated.

Technology Paper Summary.....	2
Background: The Rise of Cluster Computing	3
Background: Multi-Core Computing.....	3
Multi-Core Clustered Computing	4
Conventional Models for Multi-Core or Cluster Software Development.	4
Problems with these Conventions.....	5
Hiperware’s Solution	6
Business and Application Domains	7

Technology Paper Summary

Cluster Computing - famously used by Google, Hotmail, Yahoo and others to reliably and cost effectively run their services, as well as by government and science agencies such as NASA and the Human Genome Project to perform large-scale scientific experiments – is now ready for mainstream business.

What hinders businesses from taking advantage of multi-core and clustered hardware is the lack of a simple means to quickly develop, test and deploy enterprise software on these systems.

The increased reliability, performance and low cost of ‘ordinary’ computer hardware means that a collection of these computers can be put together to achieve significant computing capacity. In principle, the advantages to businesses are tremendous – higher computing capacity means that more data can be processed in less time, data can be processed ‘on the fly’ if required, and processing can be made more reliable and scalable to increasing demands.

A parallel emergence is that of Multi-core microprocessors (eg. Intel Quad Core Xeons) – providing in theory, the computing capacity of several CPUs in one. As vendors like Intel, AMD, IBM and Sun Microsystems start providing as many as 32 cores per CPU, multi-core architectures suddenly have the potential of providing capacity to business-computing that only expensive proprietary hardware was previously capable of.

In addition, virtualization software makes it possible to consolidate hardware by taking advantage of increasingly multi-core systems. But virtualization only allows applications must run in silos, rather than derive performance from several processors at once.

In combination, cluster computing and multi-core computers have the potential to provide unprecedented performance, scalability and reliability for enterprise software.

Yet, what hinders businesses from taking advantage of multi-core and clustered hardware is the lack of a simple means – such as a *Rapid Application Development (RAD)* method – so that

software developers can quickly develop, test and deploy enterprise software on these systems.

Hiperware presents a smart solution in the form of a pioneering software platform or ‘middleware’ written in Java, called *Hiperware Platform*. This is a generic platform, amenable to run a wide number of business-applications – including industrial

process monitoring, business reporting, business intelligence, scalable e-commerce, science and research, defense and homeland-security and other high-value, mission-critical business applications.

By taking the engineering complexity away from multi-core and cluster-computing, Hiperware Platform makes it significantly easier for developers to write software that can be partitioned across multiple computers or CPU-cores or virtual machines.

Hiperware does this by dividing software tasks across a number of computers or CPU-cores or virtual machines and then coordinating these different tasks to work together. By significantly lowering the complexity of creating such software, Hiperware Platform effectively lowers the cost, time, skill and risks involved in cluster computing – making it accessible to corporate IT-departments and to IT-Services companies.

With Hiperware Platform, developers can now focus on the business-logic rather than deal with complex engineering related to distribution, coordination and management of software tasks.

In a typical scenario, this engineering is at least as time consuming as the business-logic – by taking it away, development costs are reduced by as much as half.

Add to that savings in hardware – because expensive ‘business class’ computers are no longer necessary, and Hiperware Platform brings about unprecedented savings to business computing – calculated at between 50-80% in typical scenarios, and higher for complex applications.

High Performance/ Scalability engineering is at least as time consuming as the business-logic itself – By taking it away, development costs are reduced by half or more

Background: The Rise of Cluster Computing

Computer hardware has made exponential advances over the past two decades in relation to every important metric - features, performance (as in CPUs or graphics cards), capacity (as in memory or hard-disks), reliability, power efficiency and manufacturing methods. All of this has been achieved while continually lowering prices to end-users.

Further market access and cost optimizations have been achieved by manufacturing in cost-effective locations, by aggressive deregulation and removal of export controls on most classes of hardware, through standardization of most hardware/hardware-interfaces, through efficient supply-chain and distribution models and by the backing of better technical support.

This 'commoditization' across the hardware spectrum has led in effect to the decreased relevance of 'business class' hardware – higher-end servers and mainframes – broadly called 'enterprise computing'. This sort of computing fetches a high price premium because it hosts mission critical, high-availability or real-time business applications.

While such hardware continues and will continue to be relevant for some business applications – a reliable and cost-effective alternative can now be found to host most enterprise applications – *computer clusters*.

Computer clusters are created by connecting several significantly less expensive, yet reliable computers over a dedicated network to achieve larger computing capacity. Software is then developed to 'distribute' tasks on different computers in this cluster and communicate relevant information over the network, to achieve a collective function.

The concept of cluster computing is well-matured and used famously and in significant scale by companies such as Google – which runs a cluster of several hundred-thousand computers to host its search engine; by email-service providers such as Hotmail and Yahoo to host email accounts for millions of users;

by government to run surveillance and homeland security applications; by research organizations such as the Human Genome Project and NASA to run simulations on very large amounts of scientific data.

Large vendors such as Oracle (database servers), Autodesk (design and automation software), Mathworks (scientific software) now sell 'cluster-enabled' software applications. The market push for cluster computing is best emphasized in better cluster-licensing by Microsoft, which announced a stripped-down, 'cluster license' of Windows in June 2006; and by companies such as Apple, Hewlett Packard and Sun Microsystems which bundle software for configuring clusters of their hardware.

Even when applied to a smaller scale to include two or more computers, cluster computing brings significant value to enterprise applications. Besides the mentioned cost-benefit, cluster computing brings other essential features to enterprise-computing – *scalability* (more computers can be added to account for increased demand in computing) and *fault-tolerance* (failure of a single computer can be accounted for by designated 'redundant' computers in the cluster).

Background: Multi-Core Computing

An independent, yet related trend is the emergence of increasingly 'multi-core' CPUs. A multi-core microprocessor is one that combines two or more independent processors into a single package, often a single integrated circuit (IC)¹. A dual-core device contains two independent microprocessors and a quad-core device contains four microprocessors.

Multi-core CPUs provide in theory, the computing capacity of several CPUs in one. As vendors start providing as many as 32 cores per CPU, multi-core architectures suddenly have the potential of providing capacity to business-computing that only expensive proprietary hardware was previously capable of.

¹ Wikipedia – [http://en.wikipedia.org/wiki/Multi-core_\(computing\)](http://en.wikipedia.org/wiki/Multi-core_(computing))

However, as Wikipedia puts it, “Most application software is not written to use multiple concurrent threads intensively because of the challenge of doing so. [...] In [many] cases, multicore architecture is of little benefit for the application itself due to the single thread doing all heavy lifting and the inability to balance the work evenly across multiple cores. Programming truly multithreaded code often requires complex co-ordination of threads and can easily introduce subtle and difficult to find bugs due to the interleaving of processing on data shared between threads (thread-safety). Debugging such code when it breaks is also much more difficult than single-threaded code. [...] Although threaded applications incur little additional performance penalty on single-processor machines, the extra overhead of development was difficult to justify due to the preponderance of single-processor machines.”

Multi-Core Clustered Computing

In combination, cluster computing and multi-core computers have the potential to provide unprecedented performance, scalability and computing capacity for business software. As a common class of computing, we refer to this simply as **Multi-Core Clustered Computing**.

Despite these potential gains, the effective adoption of Multi-Core Clustered Computing for business software is hindered largely by a single factor – the lack of a simple means for software developers to quickly develop, test and deploy enterprise software.

What is effectively missing is the equivalent of a ‘Rapid Prototyping Tool (RAD)’ for multi-core clustered computing – simple yet powerful tools that would allow regular businesses to take advantage of multi-core clustered computing.

Currently, most business software development centers on ‘developer frameworks’ that do not account for multi-core or cluster computing (see following section for overview of conventional models). This means that developing business

software for multi-core clustered computers remains in the domain of the highly resourced and skilled development teams.

These development teams must understand both – the business logic underlying their software, as well as complex engineering skills specific to multi-core and cluster programming.

Much of the significant benefit evident in the ideology of multi-core and cluster computing – lower costs, higher availability and scalability - is effectively negated by the cost, time, risk and complexity involved in developing and deploying software that can run on these systems.

Conventional Models for Multi-Core or Cluster Software Development

Among business application developers, the conventional approach involving cluster-computing is to use a client-server programming model such as CORBA or J2EE and then do custom development to extend that concept to a larger number of computers.

In scientific applications, typically written in C or C++, a low-level programming interface called MPI (Message Passing Interface) and PVM (Parallel Virtual Machines) allow ‘distributed programs’ to be written. While these interfaces are powerful and reliable, their complexity has limited their usefulness to only scientific and academic communities.

Accordingly, the following models may be considered as conventional methods of working with cluster computing:

J2EE with Load-balancing

Most J2EE application servers support the concept of load-balancing. This means an incoming request (‘load’) at an application server is assigned to one among several servers – simplistic way of distribution. This is by far the standard approach in achieving some amount of parallelism in enterprise applications and is commonly used in e-commerce and online

financial services. Some commercial offerings include workflow definition along with J2EE servers.

While load-balancing works well for web-sites, it is poorly suited for Software as a Service (SaaS) or business applications – which may differ significantly in terms of their performance and scalability bottlenecks.

CORBA with load balancing

CORBA (Common Object Request Broker Architecture) is a client-server architecture which supports C/C++, Java and standards defined by the Object Management Group – IIOP (for communication) and IDL (for object representation). CORBA supports the concept of ‘Load balancing’ – this means that objects can be located on multiple CORBA servers and then communicate using a protocol called IIOP.

While CORBA has been around for more than a decade, obvious limitations such as the lack of hot-deployment, the complexity of its many protocols (and the need to learn and grasp them) means that CORBA is a dying standard – with few dominant vendors.

‘Batch Jobs’ across cluster

Some scientific and business tasks, especially those requiring offline, identical processing of large amount of data, a standard method of engaging cluster computers is to create

‘batch-jobs’. This means that the bulk-data that needs to be processed is divided into parts (equal parts as a simple rule) that are then assigned to different computers.

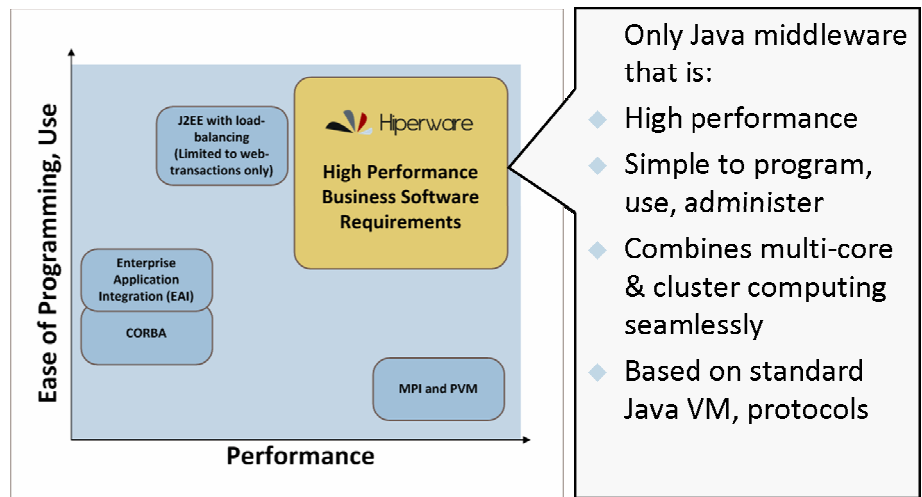
EAI (Enterprise Application Integration)

EAI is technology that enables integration of applications using common “messaging” services and protocols. These applications may run on one or more computers. Software providers such as Tibco also sell graphical workflow tools to improve usability among end users. EAI is a means for loosely-coupled applications to communicate - application-binaries must comply

with common communication protocols, sometimes using files shared on a network-mounted disk. EAI is *not* meant as a method of creating a single high-performance software to straddle across a cluster.

Grid Computing as a Related Concept

The concept of grid computing tries to create a ‘virtual’ computer cluster based on disparate, heterogeneous computers that may be located in different places. Grid computing middleware, such as Sun’s Grid Engine (SGE) or Globus, then create an accounting and administrative ‘layer’ that allows users to share this ‘virtual resource’ based on pre-determined rules and conditions. Currently, grid computing middleware is primarily used in conjunction with batch-job software and therefore popular among some scientific and academic communities.



Problems with these Conventions

Custom development on each occasion: In each of the conventional models, complex distribution of software tasks across servers or CPU-cores is achieved painfully through custom software development in each implementation.

High skills, Study of proprietary protocols: Many of these models are based on complex protocols and assume that developers have specialized training in concepts such as parallel and distributed computing, and then invest their time understanding these protocols in considerable levels of detail.

Convenience at the expense of performance: Models that emphasize on ease of use, such as EAI are at best a 'coarse' integration method. Communication is handled using high overhead, sometimes using files located on shared-disks on the cluster network or text-based protocols such as XML. EAI are therefore limited to integrating complying applications, rather than a means of developing a high-performance or scalable cluster applications.

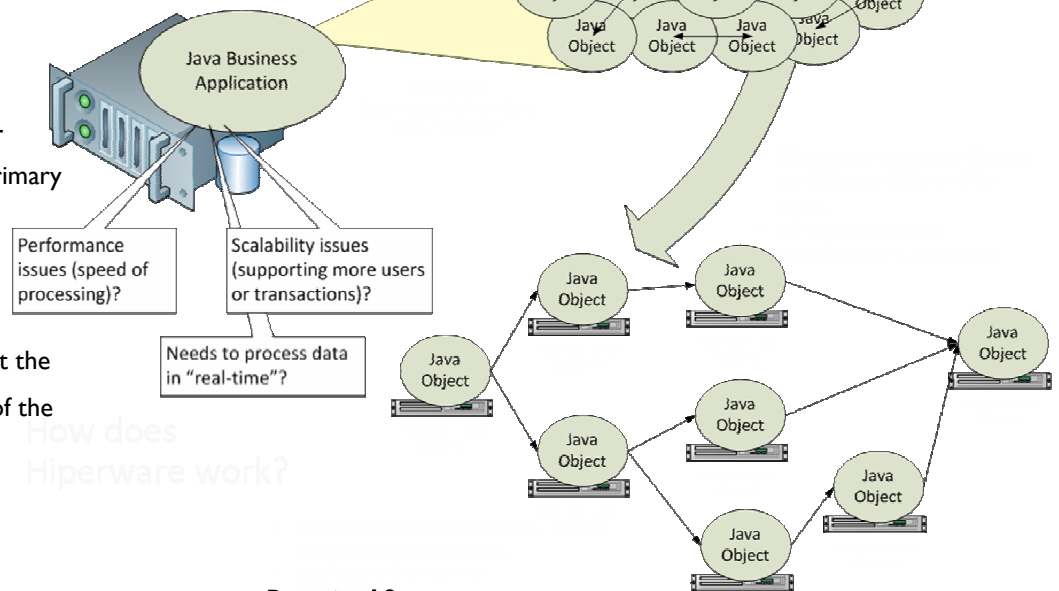
(Re)Configuration and (Re)scaling woes: To add to the complication are inherent issues when the resulting cluster application is deployed and run – Solution Architects or System Administrators may want to reconfigure the cluster, reassign hardware resources for various reasons, increase or decrease computing capacity assigned to a given software application and sometimes request for subtle changes in the operation of the software. In all of the conventional models, application-software needs to be explicitly developed to take this into account – all adding up to an expensive and time-consuming implementation.

Hiperware's Solution

The Hiperware solution lies in providing a 'middleware' - a middle-layer of software that allows businesses to develop, test and deploy software across multiple cores, servers or virtual computers – quickly and reliably.

The middleware has been designed with the following key design criteria –

1. **Developers take to it easily** – Hiperware Platform ensures this - the use of (popular) Java as the primary programming language; the use of a highly uncomplicated programming model – or API; and severely limiting the tasks that the developer must perform as part of the overall software development.



How does Hiperware work?

2. **Businesses can objectively see the benefit in terms cost and time reduction** – this is a combination of hardware cost reduction, development cost reduction, deployment/ maintenance cost reduction.
3. **Hiperware Platform does not impose a new and proprietary protocol** – instead it extends the well adopted Java standards and makes them work well on large clusters or on multiple CPU-cores.

Software Development with Hiperware:

The middleware assumes that developers are familiar with software design and development in the conventional sense.

Hiperware Platform **hides** the following complexities that developers of multi-core or clustered computer software are faced with:

1. Designing and developing parts of software to run on different computers while making sure they 'talk' to each other to achieve the overall objective;
2. Ensuring efficiency, integrity and reliability of communication between software parts running on different computers;
3. Developing software that is scalable and fault tolerant;

4. Enabling the processing of streaming data for 'real-time' applications

The process of developing an application (or re-deploying an existing application) on Hiperware Platform *involves*:

1. Developers work with Hiperware Platform by first deconstructing the application and identifying Java classes that represent CPU-intensive portions.
2. Once these classes are identified, developers develop wrappers for these classes using the Hiperware Platform API (Application Programming Interface). This means characterizing these classes by defining their inputs, outputs and run-time parameters. These inputs and outputs are defined in terms of Java-class types – this is consistent with how these classes communicate in a Single PC application.
3. Essentially, each wrapper wraps around existing Java code in the application without having to change any of the existing code itself. C/C++ or application-binaries may be similarly wrapped.
4. Once wrappers have been defined, developers deploy them into Hiperware Platform as regular Java libraries.

Making Software work on Multi-core or Clustered Computer using Hiperware Platform

1. Classes into Hiperware Platform show up in the client software called Quascade as a list of 'Components'. A Component is a visual "blackbox" representation of a class installed in Hiperware Platform, and has defined inputs, outputs and parameters.
2. The user of Quascade, typically a Solution Architect familiar with the functionality of these Components connects them together to define a dataflow.
3. Once this pipeline has been 'designed', the application can be 'run'. At this point, Hiperware Platform analyses available computer hardware on the cluster and assigns different Components to different computers in the cluster/ CPU-cores. The Solution Architect also has the

option of manually assigning certain Components to specific computers.

4. A system analyst may assist the application designer in deciding which computer resources are better suited to certain Components based on the memory and CPU requirements of those Components.
5. Components start executing in the order in which they are connected; once relevant inputs are available to a given Component from preceding Components in the pipeline, that Component is 'triggered' to execute.
6. This model is essential in real-time applications where information continuously 'streams' into the pipeline. Hiperware Platform ensures that data reliably cascades through a series of Components. While a given Component is processing one instance of data, the preceding Component in the pipeline is already processing the next data instance. This automatic pipelining of processing takes complexity away from the developer.
7. To create pipelines where data needs to be processed in parallel or collectively by several identical Components is equally trivial. This allows bottlenecks to be quickly identified in an application – and parallel processing to be used in those sections of the applications.

The model for creating high performance software is methodical and elegantly separates tasks that the software developer and solution architect must perform.

Business and Application Domains

Hiperware is working with partners to apply its technology to address a wide number of enterprise software areas.

The platform is most suitable for developing CPU-intensive software that must–

1. Process large amounts of data or simultaneous number of transactions
2. Process data or transactions instantly (*real-time*) or within difficult time constraints

Accordingly, Hiperware Platform is being exploited in the following business and application domains –

1. Business Reporting and Business Intelligence

Relevant Industries: *Manufacturing, Services Industries (banking, retail, telecom), Government, Compliance*

Business reporting and intelligence are related, yet distinct application areas. Business reporting involves capturing information from one or more sources, processing it and then summarizing information into reports designated for a given audience. Business intelligence involves analyzing business information for trends and patterns.

Currently, businesses use data-warehouses to analyze corporate data. This age-old model means that data must be first stored and organized at much expense and time – and then analyzed ‘over the weekend’. This is often too little, too late, at much cost.

What Hiperware provides is the ability to process business data as it arrives – enabling real-time decision making – a significant breakthrough for all business monitoring, reporting and intelligence systems. Alternately, it provides a complementary mechanism to data warehousing for expedited analysis, while data warehousing may be useful in subsequent, offline analysis.

Unique Requirement	Application	Markets
High-Performance	Live business reporting: monitoring, reporting, intelligence	Manufacturing, Retail, Government, Pharma
Scalability	Software as a Service (SaaS)	Online media, telecom, web-services
Real-time processing	Real-time decisions: monitoring, reporting, intelligence	Banking, Insurance, Surveillance, defense

This means that decisions can be made in response to changing business conditions – and business processes can be aligned to be more responsive and effective.

2. Web-services, Software as a Service (SaaS)

Industries: *Web & mobile application-service providers, e-Commerce, Telemedicine*

Traditional web-based applications are implemented using standard J2EE systems that use load-balancing to handle larger number of transactions. While load-balancing is suitable to scale web-sites, where transactions comprise largely of database operations, it does not apply to online software applications. Many online software applications suffer from scalability concerns, with no simple means of providing scalability.

In the high growth segment that offers broader web-based application services or Software as a Service (SaaS), there is the move for well defined consistent architectures that can account for the design and implementation of such services – broadly referred to as Service Oriented Architectures or SoA. A key issue with SoA is scalability – which must be designed depending on the specific nature and demands of the application service.

In the related telecommunications space for mobile-phone based application services, where spending is expected to grow in the USA at 29% CAGR over 5 years, this translates to

burgeoning scalability requirements for telecom services providers as well.

IDC projects that spending on Service Oriented Architectures (SOAs) will reach \$33.8 billion in 2010, from \$8.6 billion in 2006². In a Forrester Research survey of IT decision makers at 1,078 enterprises with more than 1,000 employees, SOAs have been identified by 12% as top priority for software projects in 2007 – indicating that many businesses see value in a hosted application service that may run within an intranet to minimize the licensing, administration and maintenance costs associated with desktop software licenses.

Hiperware provides a highly suitable architecture for SOA and SaaS because of its inherent ability to methodically transform an existing application into a scalable service. Moreover, Hiperware provides an unprecedented, intelligent, resource-optimized model for processing transactions – a model that may also be applied to web-sites if necessary.

3. On Line Financial Transaction Processing

Industries: *Banking, Insurance*

As with web-based applications above, current financial transaction processing systems and are currently implemented using standard J2EE systems that use load-balancing to handle larger number of transactions. Load-balancing is sub-optimal in the use of hardware resources.

When compared with other transaction processing such as web-services, financial transactions are subject to much broader analysis including financial engineering, compliance, fraud detection, prediction of trends and patterns.

Hiperware provides a suitable backend-architecture for financial transaction processing and analysis because of its inherent ability to methodically provide a scalable processing model for large number of transactions that require intensive analysis.

3. Defense and Security

Industries: Defense, Homeland security

Hiperware Platform is an ideal platform for large-scale, real-time monitoring common in defense & homeland security applications. This includes monitoring of defense data (satellite, radar, etc), other surveillance equipment (cameras, biometric devices, etc).

HIPERWARE PLATFORM is available in a wide variety of applications and available as a product that Hiperware is constantly developing, improving and is dedicated to support. The platform is available on a licensing model to VARs and qualified IT-Services partners.

Parties interested in using the software internally, or as a platform for development should contact Hiperware directly at info@hiperware.com.

² IDC Report titled 'SOA Offers Tremendous Opportunity for Service Providers'. See <http://www.gridtoday.com/grid/617472.html>