IT Production Assessment:

Azul Compute Appliance from Azul Systems

Prepared by:	Dennis Adams
Client:	Azul Systems Inc
Sponsor:	Dennis Adams Associates Limited
Document Date:	28-Nov-2005
Last Revision:	02-Apr-2006

Copyright

Dennis Adams Associates Limited

Dennis Adams MBCS CITP, AIQA, B.Sc., has over 20 years of IT experience in the area of IT Infrastructure Management, Monitoring, and Architecture and Strategy.

Based on this experience, Dennis Adams Associates Limited was incorporated in 2002, with the mission statement: -

"To Advise, Assist and Serve the IT Departments of Medium-Sized Companies, to enable them to implement Quality Production Readiness to their IT Infrastructures"

Dennis Adams Associates Limited
114 Pinner View, Harrow, Middx, HA1 4RL, UK
tel: +44 (0)7753 834 804
email: info@dennisadams.net
http://www.dennisadams.net
Registered in England and Wales, Company No. 4456808

Index

Preamble

The purpose of a Production Assessment document is to look at the practical issues relating to the Deployment and Production use of a specific piece of IT Technology. ([1])

When a new Technology solution is being evaluated, the primary focus is typically on the Business or Application Development functionality (or "functional requirements") of the product. This is very understandable, since these functional requirements are basic to the decision whether to purchase the product.

However, what is sometimes missed is the significance of the "Production-Readiness" criteria for the proposed solution (often included as "non-functional requirements"). These criteria are important since the real value of an IT solution only becomes realisable once it has been deployed into Production.

Further, it is sometimes forgotten that the costs of any solution consist not only of the initial implementation costs and customisation, but also of the day-to-day running and supportability costs. In fact, in some organisations, as much as 75% of a typical IT budget is spent on the "Support Paradigm", within which we include IT Production.

This document attempts to assess a proposed technology solution from the perspective of "Production-Readiness". It is envisaged that this type of document would provide valuable input and guidance to the non-functional requirements, and therefore assist clients in the overall assessment process.

In order to assess the product, we need to define a set of criteria against which the product should be evaluated. This list of criteria has been created on the basis of many years of IT Production experience. The key criteria that we will use to define "Production-Readiness" are:

- *Scalability*
- *Reliability and Stability*
- *Resilience*
- *Backup and Recovery*
- *Security*
- *Monitoring and Management*
- *Supportability*

Following a brief description of the product, the following sections examine each of these criteria in detail.

In addition, we discuss the key issues of Performance and Efficiency, and a brief look at alternative competitive solutions in the marketplace.

---

[1] This is one of a series of assessment papers from Dennis Adams Associates Limited looking at different IT technologies from the specific perspective of "IT Production", that is to say the management, monitoring and supportability of solutions. These papers may be distributed and quoted free of charge, providing that suitable acknowledgement is given.
For more details on the "Production-Readiness" criteria defined, and further background material, refer to the company web site at http://www.dennisadams.net

Assessment Scope And Approach

Scope

The objective of this assessment is to determine whether or not the product in question would be supportable in Production.

In order to do this, the product is assessed against the "Production readiness" criteria.

This documents is not intended to assess whether the product meets the business or development functional requirements, since these questions would be the responsibility of the IT Development or Business team(s).

Therefore, this assessment should only form part of the full review of the product.

In a more thorough assessment of a product of this type, it would be necessary to include a section on the supplier, and to consider their financial viability, and commitment to the product. This has not been included in this document at this stage.

Approach

This assessment of Azul Systems technology has been based on documentation provided by Azul Systems and questionnaires filled in by, and interviews with, Azul technical staff in the UK.

Review of the documentation and answers to questions has enabled us to build up an overview of the Azul technology. We have then specifically applied the Production-Readiness criteria to this technology, in order to assess it's suitability for Production use.

Information on software functionality has been taken from documentation supplied by Azul Systems. We understand that this functionality applies to the current versions which are generally available at the time of writing this assessment.

Target Audience

The target audience for this document is primarily IT Production Managers and Technical Support Teams. Development issues are out of scope of this document.

Introduction: Azul Systems

Azul Systems Inc ([2]) are the developers of a "network attached processing solution" for virtual machine applications, such as Java and .NET.

The concept of network attached processing has many similarities with the concepts of Network Attached Storage.

During the last few years, Network Attached Storage ("NAS") has moved mainstream, pioneered by organisations such as NetApp. Effectively, NAS enables all the small storage units that were previously attached to individual servers to be "pooled" into a centralised storage Array or Arrays. The advantage of this approach lies in the improved ability to manage Storage as a resource in it's own right, to better handle peaks and troughs of capacity and throughput, and deploy the Storage requirements more flexibly. ([3])

By contrast, network attached processing (for which we will use the acronym "NAP" in this document) is a more recent concept. NAP involves pooling the Compute power into one or more central Processing Appliances. The client's normal application servers become effectively just simple "gateways", off-loading work to the central Appliance.

Azul Systems are pioneers in this new technology area.

Potentially, NAP has similar benefits to NAS, as follows:

- Improved ability to manage CPU as a central resource in it's own right.
- Peaks and troughs of CPU demand can be managed better.
- CPU requirements can be deployed more efficiently.
- A shared resource for multiple applications.

NAP allows the CPU demands on individual servers to be pooled into a central "Compute Array", offloading the power demands from the individual servers themselves. Effectively, the individual servers just become "proxies" to the Compute Pool.

As a result, the client has an option to either replace an existing farm of mid-range power servers with lower power servers, or consolidate their existing investment, reducing operating and management costs.

[2] Azul Systems Inc. was founded in 2002, and incorporated in Delaware. It is currently based in Mountain View, California, USA. The Azul web site is http://www.azulsystems.com. "Azul" is the Spanish word for "blue". One of the founders of Azul Systems, Stephen DeWitt, was formally the CEO of Cobalt Systems, another "blue" company.

[3] It should be noted that there are other advantages of NAS including replication and server-less backup, but they are out of scope of this particular document.

The JVM Challenge

When Java was first invented, it was decided that software code would not be compiled "natively" to any particular hardware or operating system. Instead, the compiled code (sometimes called "byte code") is designed to run in an environment called a "Java Virtual Machine" (or JVM).

The JVM represents an imaginary machine architecture. In order to run Java applications on, say Linux or Windows, it is necessary to have a Java Runtime environment that maps the JVM to the specific target operating system & architecture. Once the Java Runtime environment has been written, then all Java programs would be able to run on that target. This architectural approach lies at the heart of the *"write once, run anywhere"* mantra for Java.

Obviously, if the underlying OS and architecture is relatively similar to the JVM, then the Java Runtime will be relatively easy to write, and should be very efficient. On the other hand, if there are major differences with the JVM, the result may not be as efficient. In particular, the following architecture aspects of the JVM need to considered:

- The JVM makes extensive use of Threading. Developers can create and manage threads within their own applications. User-based threading was not very common in Operating Systems until a few years ago, and some Operating Systems still do not support Threading very efficiently.

- The JVM uses a "garbage collection" mechanism for handling memory very dynamically. Most Operating Systems expect the developers to allocate and de-allocate memory as required, often in a relatively "static" way.

- Java Code is Object Orientated. Consequently the JVM is significantly impacted by memory availability. Memory needs to be available on a "heap" at very low latency, and allocated and de-allocated very rapidly. This is not always consistent with the underlying architectural model of some Operating Systems.

Ideally, then, an executing JVM should be located on an Operating System and Architecture that is specifically designed with the above features.

On the other hand, there is a requirement to be able to run Java applications on Industry-standard Operating systems (e.g. to run Websphere or Weblogic or JBOSS on Solaris Linux).

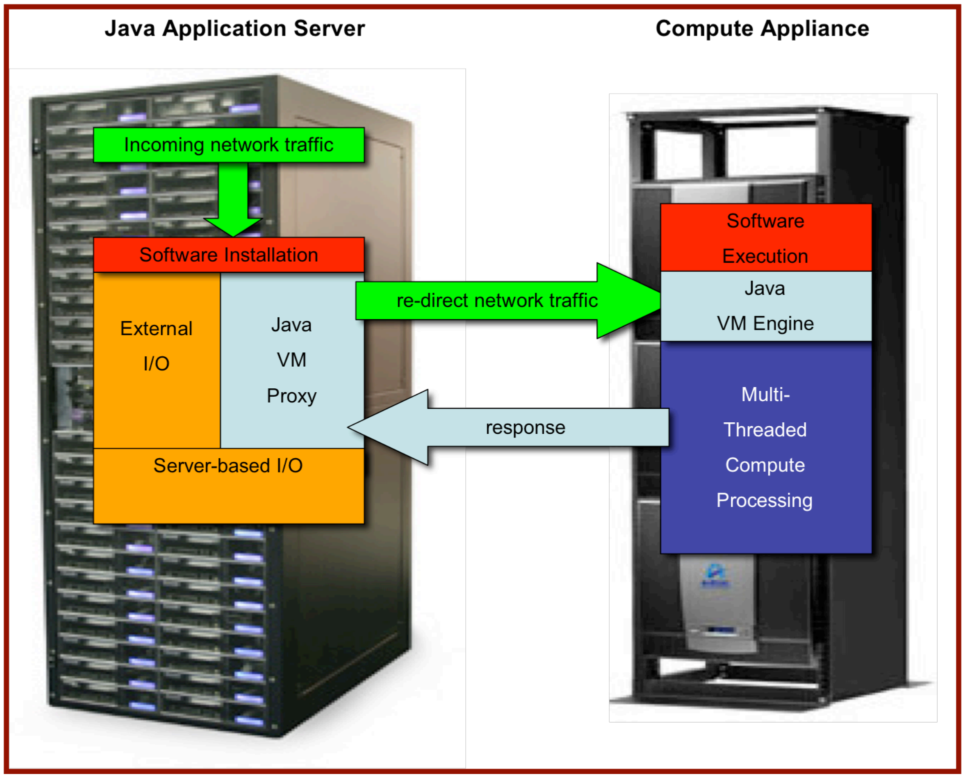
Azul Systems have provided a solution to this challenge.

Azul Systems Architecture

The Azul solution is implemented by segmenting the functionality of the runtime environment (i.e. the JVM, in the case of Java).

The "java" executable that runs in JAVA_HOME is, in fact, only responsible for the I/O activity. The actual application byte code is executed on the Compute Appliance by the "VM Engine".

A simple architectural diagram of this is shown below.



One obvious advantage of this architecture is that the "host" Operating System still appears to be performing the same functionality as before, running the business Application. However, the actual CPU processing is being off-loaded to the Azul machine.

In practice, in an Enterprise environment, there would not be a single compute appliance, but rather a "pool" of appliances, serving the compute requirements of multiple application servers.

The Azul Compute Appliance

The Azul Compute Appliance itself provides the actual computing power that is mentioned in the previous architecture section.

Azul has designed its network-based compute appliance with the single focus of optimizing the execution of virtual machine-based applications. All of the critical components have been purpose-built for network attached processing, including the Azul Virtual Machine software, the Azul Vega (TM) processor, and the coherent symmetric multi-processing architecture.

The Vega (TM) processor is a custom processor chip specifically designed for virtual machine operation. It incorporates a proprietary instruction set which is optimized for virtual machine-based applications. ([4]).

Azul's purpose-built, 24-core chip processes virtual machine workloads more efficiently than can be done by traditional instruction sets. For instance, the processors do not pause for garbage collection. It happens concurrently, as do other utility functions.

On top of the Vega architecture is the binary "operating system" which is dedicated to the efficient running of the Java Virtual Machine environments. For example, thread contention is a common cause of poor application scalability on conventional systems.

The Azul implementation of optimistic thread concurrency allows many threads to execute concurrently instead of blocking on locks. Special capabilities built into the Azul system evaluate whether actual conflict between threads occurred. If a conflict occurs, the conflicting thread is rolled back and restarted. The appliance functions as a complete "black box" - it is not necessary for any customers to have any understanding of the underlying instruction set. Modifications to application software or application servers are not required.

From the user's point of view, however, they are not aware of the internal architecture (Azul do not expose their binaries in any way), but simply that their chosen Java application is running against a different JVM. In that respect, the Azul system is a complete "black box".

[4] Azul has been able to pack 24 processor cores on a single-chip, which means that each processor is able to run 24 simultaneous parallel threads, each on a dedicated and independent processor core. Consequently, the parallel throughput of the Azul chip can be significantly higher than that of other chips currently available.

The Azul Appliance is available in a number of different forms:

Model	Processor cores	Memory	Form factor
960	96 (i.e. 4 Vega Chips)	32 or 64 GB	5 U
1920	192 (8 chips)	64 or 128 GB	11 U
3840	384 (16 chips)	128 or 256 GB	11 U

All of these models have 4 Gigabit Ethernet connections for normal use (with an option of using an additional 2 10/100 ports for out-of-band management), and the normal datacentre “RAS” features, including n+1 redundant power supplies, etc.

This is a machine designed from the CPU up to be deployed into a ‘compute pool’ to handle multi-threaded Object Oriented Garbage Collected workloads (i.e. java, with plans for .NET and other VM environments in the future) in a high-throughput way.

The Azul VM software is fully J2SE 1.4 and J2SE 1.5 compliant, is interoperable with all major J2EE application servers, such as, BEA WebLogic, IBM WebSphere, Oracle Application Server (OAS), and JBOSS Application Servers and supports popular application host hardware platforms and operating systems.

Azul Functionality

The Azul Systems solution is designed from the ground up as a Compute Appliance consisting of an array of proprietary CPUs to be deployed as a "compute pool" for modern applications. As a consequence, it is designed specifically to handle multi-threaded Object Orientated, garbage collected workloads in a high throughput way.

Some of the key features of Azul Systems offering are as follows:

Feature	Description
Architecture Support for multi-threading	The Azul appliance has been architected deliberately to support multi-threading, which is a key characteristic of Java applications. There is a very close mapping between the architecture of the java JVM and the architecture of the Appliance. As a result, the execution of JVM code is very fast.
Very large Heap Sizes	The system uses 64-bit addressing, which means that memory "heap" sizes can be significantly larger than in conventional 32-bit Java JVMs. This improves object creation, manipulation and release – key tasks in Java Applications. The Azul Systems architecture provides support for 96 GB heap size to accelerate VM performance and ensure fast, consistent user response times. This capability is available even for applications that run on 32-bit host environments.
Pauseless Garbage Collection	Garbage collection is part of the JVM architecture that often results in the Application "hanging" for very short periods of time. This is a time consuming process that is analogous to disk defragmentation. Small fragments of free memory are collected into a larger pool so they can be reallocated as requested. In conventional systems, garbage collection algorithms tend to be non-deterministic and do not scale well in heavily threaded environments especially since more threads lead to more generated garbage. Azul Systems has designed garbage collection technologies directly into the system that reduce application pauses associated with garbage collection.
Optimistic Thread Concurrency	The Azul Compute Appliance incorporates an innovative lock management scheme to improve throughput. Optimistic thread concurrency allows the processing of all threads to proceed while monitoring whether or not any write conflicts have occurred. If there are no conflicts, the writes are committed and the thread proceeds as expected. If a conflict occurs, the memory writes are rolled back and restarted after the lock contention is resolved. Because conflicts only occur in a very small number of cases, the overall effect can be a significant improvement in application throughput. Thread contention can be a performance overhead for Java applications. By changing the approach to accessing shared data, thread contention is reduced.
Compatibility with Host System.	The original application (e.g. an Application on Websphere on Solaris) still runs on the same operating system. There are no application changes required. The CPU-activity is simply off-loaded via the network to the Azul appliance.

Industry Standards for compute servers

In respect of running Java Applications, there are a number of industry standards and best practices. The most critical of these is the Java J2SE standard itself, which is fully supported by Azul Systems.

At the implementation level, Azul Systems have built a solution that segments the JVM, and executes the application on the compute appliance. As such, they are leading the market in this area, and effectively defining the standards.

Apart from Azul Systems, the compute server or compute appliance market is a very recent phenomenon. Consequently, there are no formal industry standards against which to compare the product. Indeed, in many respects we would not expect to see many standards, since an appliance should be, by definition, a "black box" which performs specific functionality on behalf of other systems.

The key standards that would be of interest to us therefore would be the interface and protocols that are used to communicate between the "client" (the Application Server, or other industry standard server which is being used to run a java program) and the "server" (the Compute Appliance itself). These protocols are, understandably, proprietary to Azul systems at this time.

In the longer term, it is possible that industry-wide standards would emerge that would cover things such as:

- Open "discovery" of compute appliances, with a form of lightweight directory service to enable any brand of compute appliance to participate in a compute farm.
- Open network protocols for the initiation, invocation and closedown of a JVM session on a compute appliance.
- Open standards for monitoring, metering, and automating the configuration of compute appliances, based on current standards.

Should these standards evolve in the future, they would enable an entire market in compute appliances to evolve, where each vendor would vie with the others for a faster and more efficient implementation.

We should stress, however, that the absence of such standards at this time should not be thought of as a barrier to adoption of the Azul solution.

As already mentioned, the most important standard in this context is the Java standard itself. By providing a fully certified JDK 1.4 and 1.5 Java runtime environment, the Azul VM Proxy acts as a genuine "black box" at that level of the application stack. Given that the VM Proxy is 100% compatible with the Java definition, customers can effectively "plug in" the Azul architecture without any need to change applications. In this sense, the Azul solution can be said to be a complete standards-based approach.

Theoretically, this approach could also be implemented for Microsoft .NET runtime environments, by writing a segmented .NET Common Lanaguage Runtime ("CLR").

Deploying an Azul Systems Appliance

One of the key features of the Azul approach is that it is designed to be deployed alongside existing technologies without any changes to existing applications or other systems. All the existing solutions of clustering, failover, session and state replication are preserved and continue to function intact.

From an application server perspective, it is only necessary to install the Azul virtual machine onto the application server host, and then change a PATH environment variable to establish redirection, and start the application. Application processing is transparently redirected from the application host to the compute pool, with no application software changes required.

Implementing such a “vanilla” solution may result in increased performance from an application, particularly at peak loading.

In practice, one benefit of the Azul Appliance is that it enables customers to downsize the power of their application server farm, by offloading the processing effort to the Azul appliance. See the section “Implementing a Compute Appliance Strategy” for a more detailed discussion of this point.

It is important to understand that not all applications would benefit in the same way from using a Compute Appliance.

For example, the Azul Appliance exhibits very high performance when executing Integer instructions and manipulation of objects, since objects are always referenced by means of an integer identifier. However the efficiency is not as marked when it is required to process large amounts of Floating Point operations.

Similarly, the Azul Appliance cannot process the use of JNI interactions with native code. All I/O operations such as file read/writes, JDBC etc. are also still performed by the original host.

These points lead us to recommend that any potential customer looking at the Azul solution should try to benchmark a representative application beforehand.

The Azul VM Proxy software is currently (December 2005) available for versions of Solaris, Linux, AIX and HP-UX. The precise versions supported are listed on the Azul website. Azul does not currently support a VM Proxy on Windows or the mainframe zOS operating system.

Azul’s VM software is a fully compliant implementation of Java JDK 1.4 and 1.5. Potential customers should be aware that if their application uses an earlier version of Java, then it might not run, unless it is upwardly compatible.

Azul have helped customers of the most common Application Servers by arranging for formal testing and certification by the suppliers. They currently have certification for JBOSS, BEAWeblogic and Apache Tomcat. Work is being done to certify Websphere 5 and Oracle Application Server.

Implementing a Compute Appliance Strategy

In a large data centre running multiple Java-based applications, it is frequently the practice to "purchase for peak", whereby systems are purchased for a running application on the basis of their occasional peak demand for CPU. As a result, each server is typically underutilised most of the time. The additional power is required only on those rare occasions when the application demand peaks.

This practice is highlighted by various analyst studies reporting overall data-center-wide compute utilization of only 10-15 percent.

Therefore, one of the real benefits of the Azul Appliance is that it enables customers to downsize the power of their application server farm, by offloading the processing effort to the Azul appliance.

The real benefit of the Azul Appliance is that it enables customers to deliver massive amounts of compute capacity and memory to virtual machine-based applications as a shared network service, similar to the manner in which network attached storage provides shared storage capacity to datacenters. This new computing model enables datacenters to achieve greater efficiencies through resource sharing, eliminate capacity planning at the application level, and enable more predictable service levels and higher reliability, while reducing power and energy costs as well as overall management complexity.

The Azul Appliance should be seen as the technology solution within an overall "Compute Management Strategy" which would involve the following:

- Deployment of the Azul Appliance(s) and integrating it into the "MOPS" (Metrics, Operational Tools, Processes and Standards) in IT Production.
- Migration of existing applications from existing hardware servers to smaller environments, where they consume less power and require significantly less computing power, since the processing work is offloaded to the Azul Appliance(s).
- Decommissioning or otherwise re-using the old general purpose servers ([5]).

The new smaller environments mentioned above could take many forms.

For example, multiple Sun systems could be migrated to small partitions in a larger physical frame. Technologies such as VMware, LPAR, VPAR etc. could be used to divide up a large system in multiple smaller "mini-servers" which would each have just sufficient power to execute the VM Proxy, but would not be required to scale further.

An alternative approach would be to use small form factor "blade" technology to host the VM Proxy.

[5] Of course, if the enterprise is growing, an Azul solution could also enable future growth utilising the existing estate of servers, without the necessity of purchasing additional hardware.

This strategy could be implemented “under the covers” in IT Production, without any changes to the application code being hosted. The Justification for such a strategy could be based on the following:

- Improved efficiency and utilisation, because of the shared compute resource.
- Continued isolation of applications from each other, including application-specific metrics.
- A server consolidation solution, leading to reduction in power and footprint.
- Reduced capacity planning effort, since compute power is centralised.
- Reduced Total Cost of Ownership and simplified Administration management.
- Ability to respond quickly to peak demands, and to move towards a “capacity on demand” model, with corresponding business benefits.
- Consistent Service Levels to end-users, irrespective of the loading.

From discussions with Azul, we understand that, in practice, much of the potential for cost savings is derived from the reduced cost of maintenance and operations. This is attributable to the fact that VM based deployments can handle the same throughput on a smaller number of machines, reducing operational cost in 3 ways:

- Reduced manpower to run the reduced number of systems
- Ability to simply deploy more compute capacity to the pool if throughput demands are exceeded with no application level changes
- Reduced Power consumption and consequent reduction on cooling requirements.

As with NAS, one of the key benefits of introducing a NAP strategy is management of capacity at a Datacenter level, rather than at a server or application level.

Scalability

Definition

One of the key benchmarks for deciding if a product is "Production-ready" is whether or not it can scale to the number of users / application instances etc. which may be required. More importantly, as the number of end-users or application instances increases, how much (proportionally) additional hardware etc. is required in order to deliver the extra capacity?

"Linear Scalability" by throwing hardware at an application is very unlikely. In practice, some near-linear scalability could be achievable. Sometimes the application architecture (or threading model) means that it works better on a single-CPU environment, rather than being able to take advantage of a multi-CPU system. In addition, sometimes applications are written with subtle in-built limitations that prevent them scaling. ([6])

As well as "Scalability", we are also interested in "expandability", i.e. to what extent can the application be enhanced and expanded to adapt to possible future requirements.

Assessment

As far as scalability of the Azul solution is concerned, we are interested in the following:

1. The ability of a single appliance to support multiple and/or high CPU-demand applications.
2. Potential for upgrading the appliance by adding additional memory, CPU and/or replacing entry-level models with top-end models.
3. The ability of the Azul architecture to "scale out" by seamlessly incorporating additional appliances into the appliance "farm".

As far as the first item is concerned, the sheer power of the entry-level appliance is impressive, consisting of 96 CPUs. The evidence we have seen suggests that system is very capable of acting as a compute appliance for high throughput applications and/or multiple applications.

Since this is an "Appliance" product, it is perfectly possible for one model to be replaced by another more powerful one with minimal impact on the datacentre, and/or for additional appliances to be added to the compute "farm".

The Compute Appliance 3840, the largest compute appliance in the product family, has, as the time of writing, more than twice the threaded compute capacity of any existing server.

The key point here is not that this is just sheer compute power. It represented **threaded** computing, i.e. the type of power which is required by a Java Virtual machine. The value of an Azul Appliance is not just measured in it's power rating. Rather, it is due to the fact that this system has been architected from the ground up to do one thing only – to run the Java Virtual Machine as efficiently as possible.

Consequently, Azul have stated that the 3840 model can simultaneously run as many as 120 applications while maintaining consistent response times for all users.

---

[6] Dennis Adams Associates web site quotes an example of where a database used a "process_id" counter, internally, to keep track of each user thread it was responsible for. This counter was, at one time, a 4-byte integer. Net result: it was physically impossible to have more than 32,768 users on the system at any one time - irrespective of the amount of hardware.

However, it is not just pure CPU power that contributes to high speed Java processing. High-speed memory access is also essential in order to run modern object-orientated applications. To address this, Azul has connected all of these processors through a passive, non-blocking interconnect mesh that enables all processors to access available memory in a given appliance.

As a result, the published benchmark work by Azul has concluded that "… despite a rapidly increasing load level, a consistent response time is achieved using the Azul Compute Appliance. The response time of the native environment degraded 5x within the load level range shown, while it remained almost constant on the Azul Compute Appliance." ([7])

Equally importantly, this particular benchmark showed a near linear response time as additional users were added to the system.

Given this evidence, we have every reason to be confident that an individual Azul appliance would scale well.

The third aspect of scalability is the ease with which new machines could be added to the processor "farm" as and when required. This is well supported in the overall Azul architecture.

The design of the VM Proxy software includes an ability to "auto discover" any available appliances on the network. This is done using a network multicast (or unicast) protocol. Consequently, the addition of a second or third appliance to a datacentre would become immediately available.

Secondly, once a VM Proxy has discovered and started to use one particular appliance, it continues to "stick" on that appliance. The only way to utilise the power of the second appliance is to restart the application itself. This may not always be practical.

Also central to scalability is the impact on the Network. The example topologies provided by Azul suggest that a Gigabit LAN with a very small number of "hops" is essential in order to exploit the power of the Azul appliance. Ideally, the client VM Proxy and server Appliance should be in the same network subnet to minimise the network overhead. There may be some situations where the network overhead itself negates any benefit from the extra CPU of the appliance.

Nevertheless, some Azul customers continue to use their existing Gigabit LAN for hosting their appliance. An alternative approach (similar to that used for SAN) is to use a dedicated switched Ethernet, and fit additional NICs into servers.

In conclusion, the Azul appliance itself is extremely scalable, and the high-end system should have more than enough capacity for most enterprise applications. Should it be necessary to expand beyond a single appliance, the Azul architecture appears very scalable, with some reservations about the amount of manual effort that may be required to manage a fully scalable Network Attached Processing ("NAP") farm.

[7] Network Attached Processing Throughput Scalability Comparison: Caucho Resin on Linux x86 Version 1.0 Benchmark report

Reliability and Stability

Definition

For the purposes of our assessment, Reliability is concerned with the extent to which the application will deliver the expected results in a consistent and repeatable fashion, irrespective of changed load and/or changed environmental circumstances.

In addition, we will define "Stability" as the ability to be able to run unattended for long periods of time without operational intervention. Reliability therefore has to do with predictable, repeatable behaviour, whereas Stability has to do with repeatable behaviour over time.

For example, some applications may be initially reliable, but their performance and/or reliability degrades over time, due to memory leaks, resulting in the necessity for the application to be restarted. This is a classic example of an application that is "Reliable" (i.e. behaves properly when it works), but at the same time it is not "Stable" (i.e. it degrades over time).

Assessment

The internal reliability of the Azul Appliance depends upon the reliability and resilience of the hardware itself.

In this respect, the Azul system conforms to normal datacentre practice, including the following:

- N+1 redundant, hot plug cooling fan modules
- N+1 redundant, hot plug power supplies
- Dual redundant gigabit Ethernet
- ECC and DRAM fault tolerance (Chipkill) on system memory
- ECC on processor internal caches and TLBs
- Parity protection on processor registers
- Memory scrubbing
- Auto de-configuration and restart around failed components

All of these factors would contribute to making the system highly reliable.

The designed mean time between failure (MTBF) for each individual Azul appliance is 40,000 hours – which equates to one failure every 4.5 years. Tests have been run which have demonstrated 11,000 hours continuous running without a failure. In practice, customers would deploy a resilient pool of appliances (see the section below).

Also, the Azul appliance has very few moving parts, which also contributes to its reliability levels.

Reliability and Stability also refer to the ability of the system to deliver the same output every time to the same operation. As previously mentioned, this is primarily a factor of the Sun Java certification. The Azul JVM is a fully certified Java Virtual machine, and therefore it's performance will be in line with the requirements of the Java certification.

One area of theoretical concern is the way in which Azul have improved the performance of their architecture by using optimistic locking to solve thread contention. In answer to this, we are given to understand that this approach is purely a matter of implementation, and is in line with the Java certification standards.

As far as Stability (i.e. with repeatable behaviour over time) is concerned, we again rely on the Java certification as the basis. However, bearing in mind the radical nature of this technology, we would also like to see some more evidence of long term "soak testing" of the appliance, to verify that there are no practical memory leaks in the implementation itself.

One of the typical long-term stability issues with Java is the tendency, in some cases, to exhibit memory leakage ([8]). This would result in application growing until it reaches its maximum heap size, at which point it could fail with a Java memory exception, or "hang". This is a application design issue, rather than a problem with the underlying JVM. Nevertheless, Azul Systems are very conscious of this practical issue, and have taken steps to ameliorate it.

The Cooperative Memory Management facility enables an application to optionally grow beyond its configured maximum heap size by borrowing memory from a shared appliance-wide "leak pool". This allows the leaking application to continue running in the short term, potentially avoiding an unplanned outage, and allowing time for corrective action. This is an excellent example of pragmatic "Production focus".

[8] Unfortunately, the myth that all memory management problems are solved by using Java, and that memory leaks are impossible is just that – a myth.

Resilience

Definition

"Resilience" can be defined as the ability to recover quickly from a failure of one or more components that make up an overall system.

Resilience therefore differs from Reliability and Stability. Reliability and Stability are concerned about behaviour under load and behaviour over time, assuming that all the components are in place. Resilience is concerned about how the system behaves if a component is lost.

A Resilience assessment takes the view that "if something could go wrong, it will do so". At a very basic level, Resilience of a Server is implemented by using dual power supplies, RAID storage controllers, Redundant Parity RAM etc. In general, it is expected that these low-level resilience issues are addressed adequately in modern high-performance production systems.

In the context of this paper, Resilience assessment is concerned with how to implement Clustering mechanisms to guard against the possibility of failure of an Operating System, and how to ensure that there is no single point of failure within the architecture.

This section should also include an assessment of how to implement Disaster Recovery mechanisms, and how to implement off-site recovery.

Assessment

Some of the Resilience aspects of the Appliance itself have already been covered in the previous section. These aspects often overlap, in that resilience of the architecture leads to reliability of the overall solution itself.

Looking at Resilience in more detail, should a memory component fail, only the affected application is terminated, and the appliance itself continues running. The memory component is removed from use automatically.

In the more extreme cases, should a memory component that is being used by the appliance itself fail, then the appliance would automatically be rebooted, with the failed component disabled.

Should a single fan or power supply fail, the appliance will continue running, and send an alert to the management console.

If the entire appliance itself fails, or a specific running JVM fails, then the VM Proxy on the client also fails as well. In that sense, the JVM is a single point of failure, and could have with catastrophic consequences for a business. In mitigation, the impact of this risk is no more or less than with a conventional JVM on a standard server. Also, the Appliance effectively isolates each JVM from the others, so that failure of one application should not impact the others.

Azul Systems have said that their Compute Appliances are never deployed in singular instances, and they they highly recommend a customer deploy them in pairs or more, to assist with potential single point of failure issues.

Being an appliance server, the network connectivity is also key to the reliability of the product. Each Azul appliance employs dual network processors (each of which has dual bonded Gigabit Ethernet data ports) for network communications and system control/service processing.

It is also possible to use multiple Network Interface Cards ("NICs") on the host servers, together with multiple Ethernet switches ("switch striping") for resilience.

Since the Appliance is literally a black box that offloads compute power, existing mechanisms for clustering of clients still work OK. The Appliance itself can also be clustered. When discovering the existence of new appliances, the Static policy server (i.e. the software that defines where the compute resources are) uses virtual IP failover. So an IP address can be moved to a new Appliance in the case of failure of an older one. However, this would require a re-start of the clients' JVMs. A "graceful failover" mechanism would be a valuable enhancement.

In conclusion, we have seen evidence that suggests a high level of Resilience both within the Appliance itself, and also within the overall Architecture Topology of the NAP solution.

Backup and Recovery

Definition

Backup and Recovery should be supported by applications and systems for two distinctly different reasons.

Firstly, Backup extends the idea of Resilience - i.e. how to respond to failure of a component - to look at how to respond to the failure of all components. This is typically implemented by using backup & recovery techniques ([9]).

Secondly, Backup can be used in order to recover the system to a known stage at a specific period of time. There may be a number of reasons for this. One reason might be that some business logic (or dependant application) has resulted in corruption and it is necessary to go back in time to recover. A second reason may be to build an "archive" or "historical" copy of the application for the purposes of analysing historical trends, or setting up a test or development environment.

"Backup and Recovery" can often has subtle implications for the design of systems. It is axiomatic that to just backup the disk contents at Operating System level is not adequate if the application is expected to perform "ACID" transactions ([10]). Therefore, it is necessary to have the ability to identify the begin-end point of business transactions so that data is consistent

Assessment

Whilst this topic is fundamental to most IT Production Assessments, in the case of Azul Systems, there is very little which needs to be considered.

The nature of a JVM is that it is a temporary copy of persistent data. The real issues with JVMs therefore, have to do with Resilience, Stability and Reliability, which are covered in the sections above.

As a network attached processing resource, the Azul appliance does not need to maintain state, and thus requires no disk drives. The fans are the only true moving part, ensuring a higher level of hardware reliability for the entire appliance.

[9] For example, failure of an entire data centre may be addressed by implementing a live standby data centre (resilience), or by taking off-site backups which can be used to re-create the application on a replacement machine elsewhere (backup & recovery). In some cases, an application may be so resilient that there is no necessity for backup or recovery for the purpose of guarding against failure.

[10] The basic properties of a database transaction are: Atomicity, Consistency, Isolation, and Durability.

- Atomicity - The entire sequence of actions must be either completed or aborted. The transaction cannot be partially successful.
- Consistency - The transaction takes the resources from one consistent state to another.
- Isolation - A transaction's effect is not visible to other transactions until the transaction is committed.
- Durability - Changes made by the committed transaction are permanent & must survive system failure.

There would be a requirement to enable backup and recovery of the configuration of an Appliance (i.e. what applications it allows min/max CPU to, security profiles, etc.). The configuration of the appliance pool is stored within the appliance itself, and replicated between all members of the pool. In production, there would be two appliances (at least) within the "compute pool'. Therefore, the configuration information would be secure, without the need for external backup. In a small organisation or development scenario, it would be advisable to have some mechanism for exporting the configuration information to an external data source or flat file. It is possible to export a domain configuration into a file, which can then be backed up in whatever CMS a customer may choose and restored at a future point.

The monitoring and metrics collection data from each Appliance is stored in a PostgreSQL database. We are not aware of any standard backup/recovery mechanisms in PostgreSQL supplied by Azul Systems. Azul are aware of this limitation. The next version of the system will deliver monitoring data via "syslog", which means that the individual backup requirements can be managed by the customer directly. In order to capture data, customers would need to parse syslog files, and insert the contents into the database of their choice.

Security

Definition

"Security" is one of those aspects of IT development and deployment that can sometimes be ignored in the early stages of software design.

In this context, we are concerned not only with the security of the application as presented to the end user (e.g. the ability to implement IP fire walls, packet filters etc.), but also with isolation of the Production Application from any development / test versions. For example, some applications have a documented API that enables any developer to call the business logic. Under those circumstances, what is to prevent a developer (either deliberately or by mistake) from calling the Live Production business logic from within an application subnet?

This section is concerned with the following basic security principles:

- *Authentication – is the person or object attempting access who they say they are?*
- *Authorisation – is the capability of this person/object clearly defined & appropriately restricted?*

Assessment

Potential security issues with an Appliance of this type come under a number of headings:

- Access to the console itself for administrative purposes.
- Control of access to appliance resources from servers.
- Isolation of individual applications JVM environments from each other.

Azul have identified the importance of security for this type of widely distributed system, and have implemented a very rich security environment. This includes both Host-based and Application-based security.

The Azul "console" is known as the "Compute Pool Manager", or "CPM". Login to CPM itself can be done using and external LDAP, "Radius", or directly. Users are allocated roles which enable them to perform specific activities within the appliance. For example, this enables administrators to be segregated from other users who may be responsible for monitoring.

It is possible to use rules to block access to the appliance by only allowing specific servers to call the JVM remotely, thus ensuring that only authorised "clients" can use the massive compute power which the Azul appliance represents ([11]).

The security rules, along with the virtual JVM architecture, also ensure that each application is isolated from the others, and is unable (deliberately or otherwise) to access CPU or memory resources which another application is using.

Access to the web-based console itself can be configured as an "out-of-band" access, and could be on an isolated network.

Azul Systems have designed the appliance very much with security in mind. For example, the systems have been deliberately "port scanned" by Azul to ensure that they have no IP vulnerability.

We have seen evidence of a high commitment to security in the Azul overall Architecture design.

[11] Without these rules, it might be possible for a developer working on a new application to access the production Appliance. This would not only represent bad practice, but would raise the small risk of a development application impacting the Production infrastructure. On the other hand, the power of the Azul system is such that, given the isolation of the VMs in the pool, some organisations can chose to deliberately allow development and production workloads to share the same hardware, with production given a higher priority. Then, for instance, developers can use spare capacity consumed by the production appliance over the weekend, again improving overall utilisation.

Monitoring and Management

Definition

Monitoring and Management is a key part of the day-to-day function of any IT Production Team.

One of the purposes of monitoring is to pro-actively identify any adverse changes in the behaviour of the system and/or it's environment, in order to take appropriate corrective action before the change impacts the business client. For this reason, "Monitoring by exception" is most appropriate. Implementing Asynchronous SNMP traps are one way of achieving this.

A second form of monitoring is "trend analysis", the purpose of which is to extract time-series data in order to model the long-term behaviour of the system and to collate it against business trends for Capacity Planning purposes ([12]).

Management is also another key role in IT Production. In this case, were are concerned with how easy it is to amend or adjust the configuration of the application, and adjust it's environmental behaviour. The important point is that such configuration should be as automated (and intuitive) as possible, in order to minimise IT Production costs for supporting the running application.

Assessment

It is refreshing to see that use of a management console has been included in the Azul Systems technology as part of the fundamental architecture.

The Azul Compute Pool Manager™ (CPM) is a fully integrated software application that enables system administrators to completely manage a set of individual compute appliances as a single pool of resources. CPM offers a consolidated view of all of the resources in a pool and allocates those resources to applications based on default or administrator defined rules-based policies.

Policy based rules enable administrators to tag CPU reservation and set allowable min and max utilisation.

The Policy-Based Management Software includes:

- Comprehensive policy-based resource control
- Real-time utilization tracking
- Metering and billing capabilities
- Seamless integration with existing datacenter management software tools
- Tightly preserved authorization and authentication security

CPM enables system administrators to gather metrics at appliance and application level. In particular, the ability to gather metrics at Application level (i.e. the Application which is actually calling the JVM via the VM Proxy mechanism) is a key requirement in the "MOPS" definition.

---

[12] The Capacity Planning role within IT Production needs to look at two aspects of IT growth:
- Trend analysis – i.e. what is the latent growth rate over time, and
- Business Metrics, i.e. how do changes to the no of business transactions correlate with the IT demands.

Azul Systems provide a Java-based application (the “Data Manager”) that interfaces directly to the metrics capturing features of the appliance. This application enables customers to download reports in XML format, which could then be used as the basis of customer charge-back and/or capacity planning.

In order to capture real-time metrics and report on them in a comprehensive way, an Event Database is required. The current recommended architecture for this is PostgreSQL on a standard server (typically, a small Linux server). Unfortunately, Azul do not provide this as part of their complete package, and we have recommended that this should be provided as an option, since many potential clients would have no prior experience of this database.

In future versions of the Azul appliance, the events are sent via “syslog”, which gives the customer the option of parsing the output themselves.

In terms of monitoring by exception and raising of alerts, SNMP Traps can be generated by the system in the case of thresholds being reached. In addition, there is a full SNMP MIB (MIB-2) for access by HP OpenView or other management tools.

In addition, to the above, Azul is investing the feasibility of supporting full integration with Tivoli and OpenView at a future date.

An external Syslog server is required for capturing syslog messages. Again, this is not offered as part of the overall Azul offering. However, this technology is sufficiently well understood by enterprise system admins, so it should not be a problem to implement it.

Azul have considered the possibility of implementing a management interface API. This will be available in a future version of the product, and will be implemented using XML over HTTP, with client libraries in Java and Perl, to enable customers to interface the management capabilities (pool management, policies etc.) to their own corporate environment. This will strengthen the capability of this product even more.

From a developer’s perspective, there are a number of tools available to help with Java Profiling, Performance Monitoring, Code Coverage, JVMPI and Bytecode Instrumentation. The Appliance comes with support for Borland OptimizeIT version 6 and Wily Technology Introscope version 5.3.1, and Quest Software JProbe. The next version of the product will be delivered with additional monitoring, instrumentation and profiling tools embedded within the JVM, accessible via a web browser.

In summary, Azul have implemented a very rich, comprehensive Monitoring and Management function.

Supportability

Definition

"Supportability" can be defined as the features which make the application or system able to be supported by a "Business as Usual" IT team. This is a general extension of the concepts of Monitoring and Mangement, above.

The significant issue with the "Supportability" assessment is whether the application can be supported at a reasonable cost. In practice, this means ensuring that we can minimise the amount of manual intervention required to keep to application at its appropriate level of activity.

Assessment

It is difficult to assess supportability without the opportunity to try out the software in practice. However, it is clear from reading the documentation that Azul Systems have put some significant thought into how their product would be deployed enterprise-wide.

The use of PostgreSQL for the Event Database has been discussed already, particularly in view of the fact that this needs to be sourced by the customers themselves. From a standards perspective, we would view this as a (minor) barrier to adoption, unless the prospective customer already manages and supports this database. This will be removed in the future versions of the product.

The policy-based Compute Pool Manager ("CPM") provides flexible central control of application resources. A compute pool with just a few compute appliances can execute on behalf of dozens of applications, resulting in increased resource utilization and simpler system administration.

The CPM appears to be an intuitive, console that enables system administrators to monitor and provision resource allocation and usage across an entire compute pool of appliances. The management interface allows customers to set application priorities and resource guarantees, and to ensure that the appliance automatically adapts to changing application workloads, based on pre-set policies.

In an enterprise deployment, it would be necessary for a number of people to have access to the CPM. Therefore it is essential that this is an intuitive product, which also contains audit trails, and role-based security. Audit trails of changes will be available in a future release of the product.

The availability of a documented API in future will enable customers to construct their own interfaces between the Azul appliance(s) and the enterprise-wide support infrastructure.

Performance and Efficiency

Definition

Although strictly outside the scope of this document, it is important to include a discussion of the Performance and Efficiency of the Azul JVM.

In this context, "Performance" is concerned with how well (typically; - how fast) the application delivers the required functionality.

"Efficiency", on the other hand is concerned with amount of resources (IT or otherwise) that are required in order to deliver the functionality concerned.

For example, it may be that an application has a very high performance, but only at the cost of a large investment in hardware and software. In this case, we would conclude that we have an application which is has a good Performance, but not very good Efficiency.

Assessment

One of the initial concerns which might be raised against the concept of Network Attached Processing is the potential network overhead of splitting the JVM into two components - i.e. the I/O activity (done on the host machine, and the CPU activity (executed by the appliance).

The effect of this has been tested extensively by Azul themselves, and by their customers. It is true that replacing a lightly loaded host JVM with the Azul solution is unlikely to give a better response time initially. However, as the load increases, the additional power of the Azul appliance more than offsets the network overhead.

Benchmark Tests suggest that this crossover point occurs when the system is executing between 4 and 8 operations per second. This is the point at which the pipelines on a traditional CPU server become insufficient, because more work is being queued than is being processed. The Azul system, on the other hand, because it is optimised for throughput computing, exhibits a virtually linear response time as the number of operations per second increases.

Published Benchmark work by Azul has concluded that "… despite a rapidly increasing load level, a consistent response time is achieved using the Azul Compute Appliance. The response time of the native environment degraded 5x within the load level range shown, while it remained almost constant on the Azul Compute Appliance." ([13])

[13] Network Attached Processing Throughput Scalability Comparison: Caucho Resin on Linux x86 Version 1.0 Benchmark report

The critical determinant, is that the network latency between the appliance and proxy server needs to be as small as possible. A millisecond latency (or better) is recommended by Azul Systems.

Obviously, all applications require a combination of Disk/Database I/O, JNI and other access, as well as pure Compute Power. Therefore, the actual performance gain (or lack of performance degredation under load) will vary with the application design. A typical application may see a 3 times improvement in throughput. Some highly processing jobs (such as a Java rules engine) have been benchmarked with orders of magnitude greater throughput.

There is considerable evidence therefore that the Azul appliance would deliver significant performance gains for applications that are currently heavily loaded. There is also considerable evidence that this performance gain would persist, even as the loading increases.

In terms of efficiency, the Azul appliance is a modern, well-designed system, with consideration given to optimum head and power consumption. For example, the 96-CPU model is a 5U high unit that dissipates only 1,000 watts of power. The largest system (384 CPU, 256 GB of RAM) dissipates 2.7 Kw.

Alternative Competitive Solutions

This section looks at some of the alternative competitive approaches or solutions that may be compared with the product being assessed.

The Network Attached Processing Marketplace

Gartner estimates that by 2008, more than 80% of all new eBusiness application development will be based on virtual machines (Java and .NET).

Consequently, there appears to be a significant market potential for any technology that is able to improve the efficiency, performance and/or total cost of ownership of Java-based applications.

In addition, the trend towards "on demand" computing can only be realised when compute power can be allocated to an application at very short notice with minimal management intervention (and subsequently reallocated when no longer required). This leads naturally to the concept of a shared "compute pool".

Such a compute pool, in order to be effective, would need to be comprised of either large numbers of CPUs, or, more likely, multi-core processors such as the 24-core Azul "Vega" chip. This assessment is reinforced by the IDC, who assess that multi-core processing initiatives such as Azul could "challenge the structure of the application server tier" ([14]).

The traditional approach to dealing with performance requirements and resilience is to architect either (1) scale-out clusters of lots of small-cpu systems, and/or (2) failover clusters of a smaller number of larger-cpu systems. These two approaches address the question of resilience, but do not necessarily provide an elegant solution to the problems of scalability. And it is the scalability issue which Azul is attempting to address – hence their tag-line of "Unbound Compute". In addition, as the Clipper Group have pointed out ([15]) both of these approaches require significant tuning and management costs.

At this stage, we consider that Azul Systems is unique in offering a solution geared to the needs of Commercial Business Transactional applications. We are not aware of any other companies with a solution that is focused on this area,

In related markets, Conformative Systems, DataPower Technology, Forum Systems, Sarvega and Tarari are all making specialist processors and hardware designed to speed up XML applications, and Clearspeed is a company in the multi-core computational processing field. Despite this, we consider that the Azul approach should be seen as a clear pioneer of this concept.

Azul have developed their solution for running Java code, but we understand that it would be technically feasible to support the Microsoft .NET runtime environment at some time in the future.

[14] "Multicore Processing: Disruption or Distraction for the IT Infrastructure?", IDC, November 2004.

[15] "Azul Systems Offloads Enterprise Java Processing – And Changes How Enterprises Do Computing", Clipper Group, Feb 2005.

Additional Impact on IT Production

Server Consolidation and Compute Appliance Strategy

As already mentioned, the Azul Solution could form part of a general Server Consolidation Strategy, during which a number of large Application Servers are replaced by an equal number of smaller systems, all off-loading their JVM CPU demand to the Azul Appliance.

As the Clipper Group have pointed out ([16]), "With the computing workloads offloaded, enterprise application servers become a bunch of front-end gateway-style mount points at which I/O, security and application patching are done."

In fact, the front-end Application Servers now need to be so small that it is possible that multiple servers could be re-hosted on a large virtualised environment. For example, a customer with 32 single-CPU machines acting as Application Servers could conceivably replace them by a single 16-way server (or even an 8-way) running VMWare, with 32 virtual instances hosted on top of it. The reduction in datacentre racking space, power consumption etc. may be a powerful ROI argument.

It is important to remember that, in order to succeed, any new strategy should include all of the elements of "MOPS", for example;

- METRICS: collection of application-based statistics for reporting to business, and for capacity planning.
- OPERATIONAL TOOLS: being the Azul appliance, syslog server, database server, and interfaces and integration with the corporate management environment, such as HP OpenView etc.
- PROCESSES AND PROCEDURES: involving the creation of a formal "CPU management" function, to allocate, organise, capacity plan and configure the NAP technologies, including support procedures and interfaced to ITIL-based helpdesk function, Service Management etc.
- STANDARDS: in this case involves the creation of a standard "build environment" which can be supplied to developers and project managers describing how new Java applications can be deployed, by default, on the enterprise-wide NAP solution.

[16] "Azul Systems Offloads Enterprise Java Processing – And Changes How Enterprises Do Computing", Clipper Group, Feb 2005.

Compute Power and "CPU on Demand"

One of the potential risks with an "unlimited resource" (or "Unbound Compute", to use the Azul tag-line), is that developers can and do find ways to utilise it fully. This is an historical trend, which is particularly obvious in the case of PC-based development.

In some cases, having virtually unlimited CPU resource could enable developers to create business applications which are hitherto impractical, and enable them significant business benefit. This can only be a positive IT contribution to the business.

In other cases, the existence of such a huge resource could "mask" inefficient or inappropriate design and implementation work. For example, developers may incorporate unnecessary threading in designs, resulting in an application that ties down CPU resource within the Appliance. Although the CPU power of an Azul machine is very cost-efficient, it nevertheless represents a real cost, which needs to be accounted for.

The way to mitigate this risk is to ensure that performance testing is included in the development / deployment life cycle for applications. This is an example of good practice, anyway. During these tests, some focus should be given to measuring the actual performance of the JVM for the application in question in order to make capacity planning predictions once the application is deployed into Production. At the same time, some thought should be given to the expected CPU demands, in order to assess whether there may be scope for application optimisation. There are various Java application diagnostic tools available on the market that can be used to assist this process ([17]).

One of the positive side effects of implementing a Compute Appliance Strategy is that it opens up the possibility of being able to deliver a "CPU on demand" paradigm. In order to do this, it would be essential that the "Metrics" element of the "MOPS" approach is implemented. Any form of "CPU on demand" must be backed up by the delivery of clearly understood and objective metrics, which would form the basis of a chargeback for resource utilisation. Azul Systems are obviously aware of this trend, and the appliance contains the necessary interfaces for extracting XML-format reports on appliance utilisation to enable customers to create chargeback reports ([18]).

[17] Another perspective of this is that the ability of the appliance to run "bad" applications well can improve the time to market for new applications and reduce the cost of developers needed to produce code. This is a "swings and roundabouts" argument that can only be resolved by prioritising the requirements of IT Development and IT Production in jointly delivering business benefits back to the enterprise.

[18] Although in practice, it is recognised that resource chargeback is extremely rare, and that many organisations do not exhibit the cultural and procedural maturity to enable them support any form of charge back. In some organisations it is possible to raise notional invoicing, which can be of significant benefit to the relationship between IT Production and the rest of the organisation.

Conclusions

The Azul Compute Appliance represents a revolutionary approach to The JVM Challenge described in the Introduction.

However, unlike many revolutions, this is totally compatible with the existing infrastructure investment and best practices. An IT department that has deployed an Azul Compute Appliance simply has to replace the existing JVMs with the Azul VM software in order to take advantage of the "Compute Pool". Apart from this change, existing applications would continue to run on their host hardware without any amendment whatsoever.

The Azul Appliance is therefore one of the first of the new generation of Network Attached Processing ("NAP") solutions, which we would expect to see deployed in datacentres in the future. Just as NAS has become a de-facto technology in IT Production, so we would expect NAP to have a similar high profile in the future.

As far as the production worthiness of this solution is concerned, it is clear that Azul has taken a great deal of effort to ensure that their product has been launched as a "datacentre ready" product from the beginning (rather than, as sometimes happens, trying to build a proof of concept offering before "hardening" it later). This is evidenced by purely technical choices such as additional redundant power supplies etc.

From the Scalability perspective, the entry-level machine has massive computing power, and represents a very powerful high-end system. Beyond that, the scalability is based around the architecture of the "Compute Pool", managed by the Compute Pool Manager software.

The systems come with a very high designed MTBF. IT Production managers can have a great deal of confidence in the Reliability and Stability of this solution.

Resilience is also addressed very well, not just internally within the product, but also in the provisioning of "discovery" and other scale-out features. Currently, there is no "graceful failover" clustering solution, but we are hopeful that this would be available at some time in the future.

Backup and Recovery are not an issue for a Compute Appliance. On the Security side, there is a very fine-grain security architecture & use of roles to ensure access control and application isolation.

As far as Monitoring and Management is concerned, some customisation/implementation would be required by the customer in order to integrate the Azul Appliance(s) into the Enterprise Management and Monitoring framework. The future API promises to make this activity even easier.

Monitoring data is available via an XML-based interface, with makes for relatively easy integration with other Production infrastructure.

In conclusion, the Azul Appliance is a very powerful tool that promises to improve the management of Compute Power in the datacentre, and provide a valuable impetus for cost-effective datacentre consolidation. It is very much a "production-ready" solution for the 21st century datacentre.

Assessment Matrix

The Main conclusions of this assessment can be presented in a tabular format, as below:

Production Criteria	Theoretical Assessment	Test Results	Notes
Scalability	5	N/A	High-end and highly scalable solution.
Reliability and Stability	5	N/A	Reliable production-ready solution.
Resilience	4	N/A	Datacentre ready technology.
Backup and Recovery	5	N/A	backup of all config files provided.
Security	5	N/A	Security roles & Application isolation.
Monitoring and Management	4	N/A	Some monitoring integration required.
Supportability	4	N/A	Supportable with minimal training.

In most cases, it will be necessary to combine this Theoretical Assessment with specific Tests to identify and prove that the application meets the criteria mentioned. In the case of Azul Systems, we have not been able to do so because of time constraints. Therefore, there are no test results in this document at this stage.

The meanings of the scores are as follows:

1. Application or System is considered to be totally unsuitable for IT Production use. Costs of support are likely to be prohibitively high if the application or system were ever introduced into IT Production.

2. This Version of the Application or System is considered to be unsuitable for IT Production use, but could be used for software development, and additional discussions with the vendor should be held in order to introduce required features in a future version. Costs of support are likely to be very high if the application were ever introduced into IT Production.

3. Application or System is recommended for deployment into production with some additional customisation required by the client or vendor in order to improve supportability. Costs of support are likely to be in line with costs for other applications of this type.

4. Application or System is suitable for Production deployment, with very little additional customisation required. The client can implement any such customisation, without any necessity for involvement from the vendor. Costs of support are likely to be in line with costs for other applications of this type.

5. Application or System is suitable for Production deployment, with minimal customisation. The vendor has demonstrated a strong understanding of the principles of "Production Worthiness", which are reflected in the design and implementation of the product. Costs of support are likely to be in line with, or less than, costs for other applications of this type.

In addition to the above, we have also considered the topic of Performance and Efficiency. These are criteria that we are unable to assess properly without the opportunity to conduct proper testing.

However, given the high scores that Azul Systems have gained in the other sections, we have good reason to believe that this would be an Efficient and High Performance solution which would be very suitable for deployment in an IT Production environment.

Sources

This document has been prepared from the following information provided by Azul Systems:

Document Name	Details
Reliability, Availability, and Servicability	RAS characteristics of the Azul Compute Appliance Version 1.0 Whitepaper.
Network Attached Processing	Scaling Enterprise Application Deployments Version 1.0 Whitepaper.
Network Attached Processing Throughput Scalability Comparison	Caucho Resin on Linux x86 Version 1.0 Benchmark report
Reducing the cost of application-tier deployments with network attached processing	Network Attached Processing TCO Version 1.2 Whitepaper
Industry Validation	Early Customers experience
Platform Interoperability	Virtual Machine Software Platform Interoperability as of September 2005
Unbound Compute for Enterprise Developers	Details of Compute Appliance Model 960B.
Compute Pools: Unbound Compute for Enterprise Applications	Details of Compute Pool technologies

Further information can be obtained from the Azul Systems web site at http://www.azulsystems.com

Information from these documents was used in preparing interview questions that were put to Azul Systems Technical team in the UK.

In addition, the following Analysts reports were reviewed in the preparation of this assessment:

Document Name	Author	Date
Azul Systems Offloads Enterprise Java Processing – And Changes How Enterprises Do Computing	Clipper Group	Feb 2005
Multicore Processing Scenarios, 2005-2009: Disrupting the IT Market in Three Generations?	IDC Technology Assessment	Jul 2005
Multicore Processing: Disruption or Distraction for the IT Infrastructure ?	IDC Insight	Nov 2004
2005 Preview - Systems	The 451 Group Market Insight	Dec 2004

About the Author:

Dennis Adams has over 20 years experience with Managers of IT Production Systems, and now provides a specialised Consultancy Service to help IT Production Managers take a more Strategic view of IT Production Management.
He can be contacted via the company website at http://www.dennisadams.net