SCALABILITY IN COMPUTING FOR TODAY AND TOMORROW

Swati Sharma, Priyanka Madnani, Nikki Kumari
Department Of Electrical And Electronics Engineering
Dronacharya College Of Engineering,Gurgaon

***Abstract*- Achieving scalability in computer systems without sacrificing Liability requires a synergistic combination of system architecture, fundamental technologies, and implementation The art of building and running a super-scale compute cluster is complex and challenging. When growing a cluster into the super-scale range, issues that are minor or go unnoticed in a smaller cluster suddenly become glaring issues. What works at small scale offers no promise of working at super-scale levels. Future requirements are beginning to emerge that are expected to require a three orders of magnitude increase in computing speed and associated infrastructure. Scalable systems are extremely dense in terms of both computing power and storage capacity; that is, they are able to put a lot of computing power and storage capacity into each cubic foot of data center space they occupy. Not all dense systems are scalable however; many require a data center to "start big", and then to increase computing power and storage capacity in large increments. Scalable systems operate differently, providing minimal configurations and then allowing scale out operations in small, affordable increments that match performance and capacity demands. Because of this capability they allow a data center to start small and then to scale out to develop a massive storage infrastructure capable of supporting even the most data-hungry applications. Ideally, performance and storage capacity should scale independently of one another, allowing the system managers to add performance without spending money on added storage capacity, or to add storage capacity without the necessity of budgeting for unneeded extra performance. Systems scale as needed, in an affordable and predictable fashion.**

***Index Terms*- DRM, DRMAA, NAS**

I. INTRODUCTION

As the problems being tackled by researchers and enterprises grow in complexity, the tools being used also must grow in power. At the leading edge of the line-up of power tools in a researcher's arsenal are super-scale compute clusters. With tera FLOPS of compute power, peta bytes of storage, and terabytes of memory, these computing giants make possible computations and simulations that would otherwise take orders of magnitude longer to run or be out of reach altogether. Future requirements are beginning to emerge that are expected to require a three orders of magnitude increase in computing speed and associated infrastructure.

Several easily identifiable trends are causing the increase in both personal and corporate data. Personal data (much of which is maintained at corporate sites) sees larger file sizes from applications, cameras, and cell phones. More users watch and share streaming video. Broadcast and social network sites demonstrate dramatic growth.

Storage is being consumed faster than ever by private users, and is being consumed just as dramatically by business users, particularly in the following areas:

1. **Increased use of rich media by mainstream applications**. :- Rich media has proven to be extremely useful in mainstream corporate applications. Examples of this are streaming video embedded in computer-based training programs, webinars that are stored on corporate websites for later use, and the like. Customer-facing Web applications in particular often require substantial amounts of data and high-performance storage to house them.

2. **Web 2.0**. :- As the Internet increasingly becomes a platform for mass collaboration and computing, the rich applications that support these operations may well place the greatest demands on storage. It remains to be seen where "Web 2.0" will eventually go, but Web syndications such as RSS and numerous other applications that transfer data require more than just bandwidth; there is also a requirement for a place to store whatever the bandwidth has transferred between one collaboration node and another.

3. **Digital archiving**:- For a number of reasons, most companies find that it makes good sense to convert their analog archives (usually consisting of myriad boxes, cases and cartons of documents of all sorts) into digital archives. Such data is typically scanned and then stored on digital tape. Storage costs are less of course, but the real reasons behind this shift have much more to do with satisfying a need to access and identify off-line data in a timely fashion. Properly filed and accessible data, even old data, is part of a company's intellectual property and should be viewed as an asset to be leveraged whenever possible.

4. **More granular data**:- Many applications have increased the granularity of the data they use. The most obvious example of this is the digital camera, which five years ago provided pictures in a range of one-to-two megapixels, and which today creates consumer photographs of eight megapixels, 10 megapixels, and beyond. Similar increases in granularity are seen in the medical and scientific fields (with significantly more accurate CT scans and digital x-rays), and, perhaps most dramatically, in commercial television, where the shift to digital (and beyond that, to high definition) formats now means 13.5 GB of storage space is necessary for each hour of encoded video. Even more traditional applications such as databases are making use of new technologies such as visualization and data cubes which can add substantially to their storage requirements.

Whatever the cause for such growth, the storage must be managed efficiently so that the data is adequately protected and always available whenever and wherever it is needed.

II. SUPER-SCALE COMPUTING

Super-scale computing is the application of very large compute clusters to the task of solving computational problems. A compute cluster is a collection of computing resources that work together to solve a problem or a set of problems. The cluster usually is federated through some sort of software middleware, such as a parallel library such as the Message Passing Interface (MPI), a **distributed resource manager (DRM)** such as Oracle Grid Engine, or an application framework like Apache Hadoop. In typical applications, a compute cluster can range in size from a few nodes to a few hundred nodes. For most application needs, compute clusters in the range of tens to hundreds of nodes are sufficient. However, the size of the data or the scale of the computations for some applications is such that significantly more compute power is required. These applications need super-scale compute clusters with hundreds of nodes and thousands of cores. Such massive compute clusters are expensive to build and maintain, but provide a level of computational horsepower found nowhere else in the world. Published twice a year, the Top500 Supercomputer Sites is a list of the most powerful systems. As of June 2009, the world's fastest super-scale compute cluster is the Roadrunner system installed at the U.S. Department of Defense. The cluster is composed of 278 refrigerator-sized racks containing 18,802 processors with a total of 122,400 cores that occupies 5,200 square feet1. The systems are connected by over 55 miles of fiber optic cable, using both Infinite Band and Gigabit Ethernet technology. The entire cluster—including cables—weighs over 500,000 lbs, consumes 2.35 Megawatts of power, and has a maximum computing potential of over 1 Peta-FLOP (over 1,000,000,000,000,000 operations per second). For comparison, a modern desktop workstation with two quad-core Intel® processors offers approximately 100 Giga FLOPS of peak performance. The precise definition of super-scale is still open to debate. The general consensus is that a super-scale cluster contains more than 1,000 nodes or more than 10,000 cores, or delivers more than 100 Tera-FLOPS of performance.

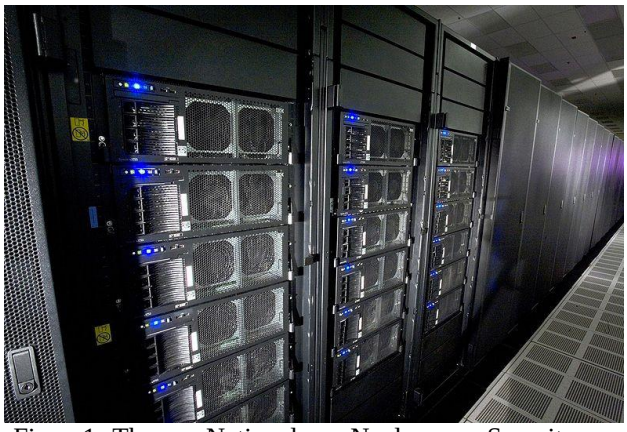

Figure1:-The National Nuclear Security Administration's Roadrunner Cluster

1. USERS USING SUPER-SCALE COMPUTING:-

The most common use for a super-scale compute cluster is research, from high-energy physics to meteorology and astronomy. The largest clusters on the Top500 List are used by researchers to perform calculations and process data that would otherwise take weeks, months, or even years to do on less grandiose computing resources.

The Texas Advanced Computing Center's Ranger cluster is a good example. Built through a $30,000,000 grant from the National Science Foundation in 2007, the cluster was targeted to be the world's fastest supercomputer at over a half Peta-FLOP in peak, theoretical performance when it launched. In the June, 2009 Top500 List, the Ranger cluster was listed at number eight. The cluster is part of the Tera-grid, a collection of computational resources distributed across the U.S. through member organizations. (When it came online, the Ranger system was more powerful than all of the other Tera-grid systems combined.) Researchers with permission to use the Tera-grid can leverage its enormous compute power to answer open questions in their research. One notable use of the Ranger cluster through the Tera-grid has been the Pacific Northwest National Laboratory project on video facial recognition. Ron Farber and his team created an application that ran distributed across 6o,000 CPU cores on the Ranger cluster to perform real-time face recognition in videos with 99.8% accuracy. Because of the sheer volume of video data being created every moment, the enormous scale of the Ranger cluster was important for testing the scalability of the recognition algorithms.

Figure 2. The Ranger cluster at the Texas Advanced Computing Center.

Another example is the EINSTEIN cluster at the Naval Oceanographic Office Major Shared Resource Center (NAVO MSRC), which debuted at number 26 on the Top500 List with over 12,000 processors. The NAVO MSRC is best known for the Navy Operations Global Atmospheric Prediction System (NOGAPS), which produced the most accurate five-day prediction of Hurricane Katrina's behavior. Today, the EINSTEIN system is used for modeling global weather patterns for prediction. A popular commercial super-scale compute cluster example is the Google compute cluster. While details are confidential, it is estimated that Google uses approximately 450,000 production servers that are distributed across at least a dozen datacenters throughout the world. Instead of simulation or modeling, Google's compute clusters are the engine behind the popular search tool. They also are used to power offline data mining jobs against the many terabytes of data Google has archived.

2. USES FOR LARGE SCALE CLUSTERS:-

A variety of disciplines can take advantage of large-scale compute clusters. In the examples below, the applications themselves are not highly scalable—but thousands of smaller jobs must be scheduled, resources allocated, and the execution managed.

2.1 Image comparisons:-

Image recognition is gaining in popularity, and poses interesting challenges for compute clusters. For

example, consider a large database of images that need to be compared. Some kind of image processing algorithm must be used. If each application is single-threaded, then thousands of jobs would need to be submitted into queues. With potentially millions of images needing to be compared to thousands of source images, it is imperative that the DRM software be able to scale and handle these massive loads.

2.2 Image rendering:-

The software rendering of images, particularly those seen at the movies or on TV, can be an enormous computational and scheduling challenge. For example, a two hour long movie requires 216,000 frames to be rendered (30 frames/sec x 60 sec/minute x 60 minutes/hour x 2 hours), with each frame undergoing multiple rendering phases. With so many frames needing processing, some kind of job management for final rendering is critical.

2.3 Molecular modeling:-

Simulating millions of compounds and analyzing how they react with other compounds requires
enormous computing resources. Since many of these jobs can be run at the same time, and
independently of one another, compute resource management is critical for both throughput and cluster utilization.

III. MASSIVELY SCALABLE NETWORK ATTACH STORAGE

Massively or extremely scalable NAS is storage capable of accommodating file-based content that is always growing and that must be pervasively available. Furthermore, such devices must do this in a way that is affordable in terms of a device's initial cost, the incremental costs associated with each scale out operation, and of course, the mix of daily expenses that compose the OPEX. All storage must be fully visible all the time both to the clients accessing the data and to the management console that monitors the system. The system should be rapidly scalable, and should be capable of moving any data to an application in both small increments and very large increments (hundred-plus terabytes at a time).

Extremely scalable NAS should offer advantages over more traditional storage in the areas of scalability, manageability, and affordability.

3.1 Scalability. Such systems are extremely dense in terms of both computing power and storage capacity; that is, they are able to put a lot of computing power and storage capacity into each cubic foot of data center space they occupy. Not all dense systems are scalable however; many require a data center to "start big", and then to increase computing power and storage capacity in large increments. Scalable systems operate differently, providing minimal configurations and then allowing scale out operations in small, affordable increments that match performance and capacity demands. Because of this capability,

they allow a data center to start small and then to scale out to develop a massive storage infrastructure capable of supporting even the most data-hungry applications. Ideally, performance and storage capacity should scale independently of one another, allowing the system managers to add performance without spending money on added storage capacity, or to add storage capacity without the necessity of budgeting for unneeded extra performance. Systems scale as needed, in an affordable and predictable fashion.

3.2 Manageability:- The maximum value of massive scalability – particularly in on-demand environments -- can only be realized if the newly added content and the devices that hold it can be managed easily and with efficiency. This means several things. Wizards and/or policy-based decision making are fundamental. Scaling operations, no matter how large they may be, must never impose substantial planned downtime. Initial deployment and day-to-day management of both the storage and server components should be done through a single, unified interface that streamlines all aspects of provisioning and configuration, and which allows servers and storage to be scaled both together and separately, as requirements demand. Adding additional storage modules or server blades must be a simple, automated affair that can be handled with relative ease by any level of staffer. Advanced systems using more sophisticated management tools -- high avail-ability (clustering) software for example – should have those tools integrated within the management console. Systems with capabilities such as these will enable even relatively inexperienced administrators to scale their management capabilities as well,

eventually enabling them to manage Peta-bytes rather than terabytes of storage.

3.3 Affordability.:- Clearly, massive scalability and highly available systems will be of little use to IT managers if they are not also affordable. Systems should have no hidden costs, should scale out in a granular and affordable fashion, allowing the system manager to invest only where investment is required, and should be managed by a single interface that addresses both performance and capacity management. A single easy-to-use management interface such as this has the potential to drastically lower administrative expenses.

IV. SCALABLE SHARED MEMORY MULTIPROCESSING

A scalable SMP (SSMP) restructures the SMP class to incorporate the advantages of the other architectures while retaining the programming model and low latency communication of the SMP.

To understand the structure of an SSMP, one should begin by looking at the SMP in more detail. The bus structure of the SMP is key to its tight integration and cache coherence, but also to the scalability limit of the system. The cost of the bus itself limits how small a system can effectively configured. The fixed bandwidth of the bus limits how far the SMP can scale to support a large number of processors. The first step in the evolution of the SMP is to replace the bus with a switch. The switch removes the bus bottleneck by giving the system scalable bandwidth that can grow as the system grows. Further, the switch can be small to reduce costs when the system is small, and grow as the system grows. In conjunction with this change, the cache coherence mechanism must be changed, since the snoopy schemes used on bus-based SMPs rely on broadcasting every memory reference to all caches. The requirement for cache is removed by adding directories to memory so that the memory knows which processors hold a copy of a particular cache block and can send point-to-point messages to keep caches coherent. Replacing the bus with a switch removes the bus bottleneck, but is still is not ideal because

the switch still adds latency and uses shared bandwidth for memory locations that are accessed only by a single processor. The solution to this is to push portions of the memory through the switch and distribute the shared memory and I/O among sets of processors. With a distributed shared- memory, or DSM, structure; memory and *U 0* that has affinity to a given set of processors can be accessed with lower latency and does not use the shared bandwidth of the global interconnect. Memory bandwidth increases naturally as processors are added. Further, modularity is greatly increased because each "node" is a complete functioning unit and an entry system need not have a global switch at all. DSM machines are also referred to as NUMA or non-uniform memory access machines. This is in contrast to traditional bus-based SMP or switch based PVP system where all memory is equally distant, providing uniform memory access. NUMA architectures that support caching of local and remote memory are referred to as CCNUMA, for cache-coherent NUMA. The DSM or CC-NUMA system has the same basic structure, and thus scalability, as an MPP or workstation cluster. The primary difference is that all memory is accessible and cacheable by all processors. Furthermore, all *U 0* can be accessed by any processor and *U 0* devices can DMA directly into any portion of memory, as in an SMP. All that has changed from the switch-based SMP is that memory and I/O resources have been distributed along with the processors.

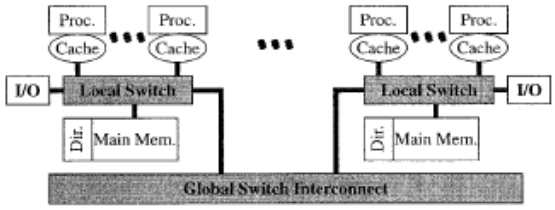

Figure 3:-Distributed shared memory system

2.1: DIRECTORY BASED-CACHE COHERENCE

Directory-based cache coherence is key to SSMP because the only processors involved with the coherence operations for a given memory block are those that have previously read or written that block. Thus, the overhead of cache coherence is never more than a fraction of the traffic required to access the memory block if it were never cached at all. Furthermore, memories and directories only interact with processors, never with one another. Thus, bandwidth to global cache coherent memory can be scaled with directories by simply adding additional

memory banks, or in the case of the DSM or CC-NUMA systems, by adding additional nodes to the system. Directory-based cache coherence was originally proposed by Censier and Feautrier in 1978. However, during the 1980s, commercial SMP systems were based on snoopy cache coherence because snoopy schemes were simpler and placed the burden of coherence on the caches themselves. Renewed interest in directory-based cache-coherence began in the academic community in the late 1980s when the inherent bottlenecks of bus-based SMP systems began to be felt. A few research groups built prototype machines to explore the performance and design issues with these scalable systems. Two of the major efforts were the DASH project at Stanford[9] and ALEWIFE at MIT. Another early effort at a directory protocol implementation was taken on by the IEEE Scalable Coherence Interface working group. This group defined an interface standard for modules that includes a directory- based cache coherence scheme that could be used to build up SSMPs out of nodes conforming to the SCI standard. The earliest commercial based DSM systems were the Kendall Square Research KSR- **I ,**
introduced in 1991; and the Convex Exemplar, introduced in 1993. The Exemplar was based on the IEEE Scalable Coherent Interface protocol. These early machines were not very successful with KSR eventually folding; and Convex struggling financially and eventually being acquired by Hewlett-Packard. The limited acceptance of these early DSM machines was due to improved bus technology which yielded bus-based SMPs with more than a gigabyte per second of memory bandwidth, and to the fact high-performance switches could only be built with expensive bipolar or gallium arsenide technology at this time. Today, the need for higher performance and greater scalability has renewed interest in DSM systems. Technology improvements in commodity CMOS have also made these systems much more cost-effective. Some of the announced DSM systems include SGI Origin servers, Sequent's NUMA-Q systems and Data General's NUMA Line. Convex in conjunction with HP has announced their 2nd major generation of Exemplar DSM systems, the X-class. The usability of a DSM system as a scalable SMP depends on the latency and bandwidth to remote memory. If the system can achieve high bandwidth and low latency to all of memory, then it can function as an SMP. If latency is very high, or the bandwidth is very low, then use of remote memory needs to be carefully controlled by the user, causing the system to function as more of a distributed memory system with a shared memory communication mechanism than a scalable SMP. For example, let's assume that the processor is stalled on cache misses, waiting for memory *25%* of the time. For this case, Figure ***2*** shows a plot of relative efficiency based on the ratio of local references. This plot demonstrates that if remote access times are kept within 1.5-3x local; then even when all references are remote, efficiency is still two thirds of when all references are local. If locality can be increased to 50% or better, then efficiency is above 80%. In contrast, if remote latencies are 5-10 times local, then locality must be kept very high or performance falls off dramatically. Similarly, if bandwidth to remote memory is only a fraction of that to local memory, queueing delays can increase latency and pull down efficiency when remote memory is accessed. Figure 5 illustrates this effect for the case in which local memory bandwidth is 40% utilized when all references are local. This graph shows that for lower ratios of remote to local bandwidth, memory occupancy is higher for an equivalent ratio of local to remote references. Ideally, remote bandwidth equals local, and memory utilization is unchanged by locality.

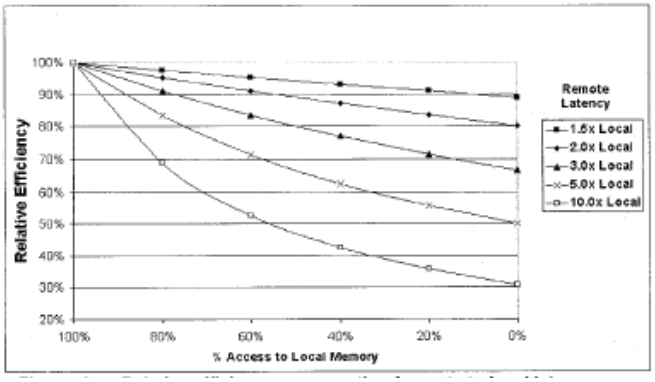

Figure 4:- Relative efficiency versus ratio of remote to local latency

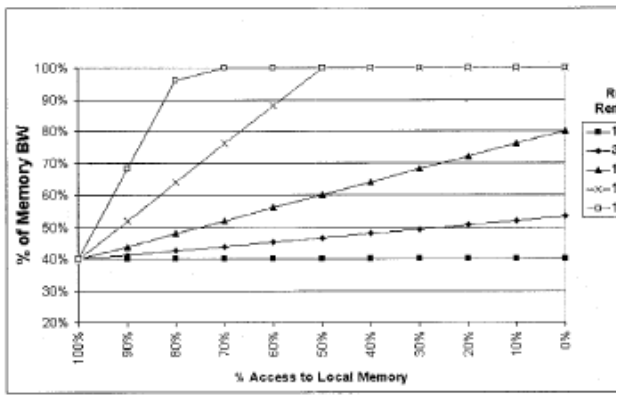

Figure 5:-Memory utilization versus ratio of remote to local bandwidth

V. SOFTWARE ENGINES :-

5.1 Oracle Grid Engine:-

The Oracle Grid Engine software is a distributed resource management system that fosters increased utilization, better workload throughput, and higher end-user productivity by harnessing existing compute resources. By transparently selecting the resources that are best suited for each segment of work, the Oracle Grid Engine software is able to distribute workloads efficiently across a resource pool while shielding end users from the inner workings of the compute cluster.In order to facilitate the most effective and efficient scheduling of workloads to available resources, the Oracle Grid Engine software gives administrators the ability to accurately model as a resource any aspect of the compute environment that can be measured, calculated, specified, or derived. The software can

monitor and manage resources that are concrete, such as CPU cores or system memory, as well as abstract resources, such as application licenses or mounted file systems. In addition to allocating resources based on workload requirements, the Oracle Grid Engine software provides an arsenal of advanced scheduling features that allow administrators to model not only the compute environment, but also the business rules that control how the resources in that compute environment should be allocated. The architecture of an Oracle Grid Engine cluster is fairly straightforward. A central master component, known as the Q-Master, coordinates communications for the cluster and is responsible for scheduling workloads on available resources. Optional Shadow Daemons monitor the state of the

Q-Master and can start a new Q-Master if the current master process or node becomes unresponsive. On each compute resource, an Execution Daemon manages the execution of workloads on the node. Users interact with the cluster via command-line tools, graphical user interfaces, and Distributed Resource Management Application API (DRMAA)-enabled applications. All activity in the cluster is logged into an accounting database that can be accessed by users and administrators through the Accounting and Reporting Console (ARCO).

10

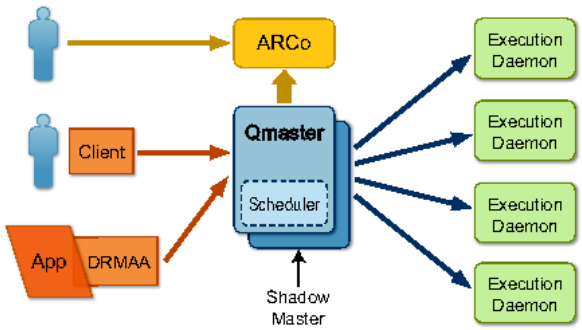

Figure 6:- The Oracle Grid Engine software architecture

Oracle Grid Engine software includes the Service Domain Manager, software that enables the sharing of resources among separate clusters . Through the Service Domain Manager software, an administrator can configure service-level objectives to govern service levels in the managed clusters. As load in one cluster increases beyond the boundaries set by the service-level objectives, resources from less heavily burdened clusters can be automatically redeployed to the overloaded cluster until the overage has passed. Should no free resources be available locally, the Service Domain Manager software also has the ability to provision resources from a compute cloud provider, such as Amazon's Elastic Compute Cloud (EC2), to add to an overloaded cluster. When the overage ends, these resources are returned automatically to the cloud. In addition, the Service Domain Manager software is able to remove unused resources from managed clusters and place them in a

spare pool of resources. Resources in this spare pool optionally can have power management applied to reduce a cluster's overall power consumption during off-peak
periods.

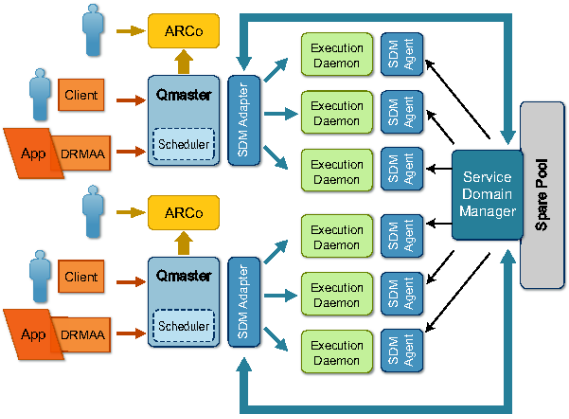

Figure 7:-Service Domain Manager

1.2 BitSwarm Engine 3.0:-

One of the key performance element in SmartFoxServer 2X is the core network engine, codename BitSwarm, currently at his 3.x version which provides several advantages over most of the other competing game servers.

BitSwarm is specifically designed with massive multiplayer games in mind and provides a remarkable edge over the usual all-purpose socket libraries employed by other server products (typically Apache Mina 1.x).

In essence BitSwarm provides SFS2X with TCP/UDP connectivity, session management, network security tools, the HRC (Highly-Resilient-Connections) system, pluggable HTTP tunneling, monitoring and more, using an highly scalable non-blocking design.

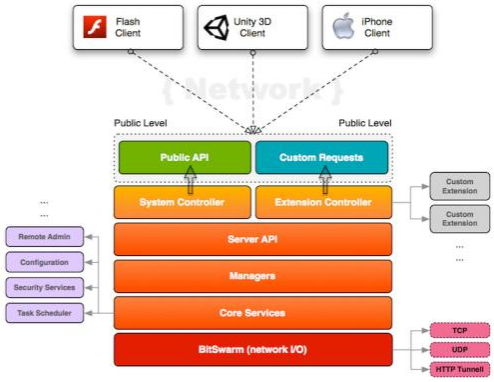

Figure 8 :-Bit-Swarm Engine architecture

VI. CONCLUSIONS

The Oracle Grid Engine software is a highly scalable and flexible system for managing workloads on a wide variety of clusters, ranging from tiny single- or dual-node clusters to huge super-scale clusters like the TACC Ranger cluster with almost 4,000 hosts and over 63,000 processor cores. While the Oracle Grid Engine software is capable of scaling to meet the largest super-computing challenges, some care must be taken in cluster construction and configuration to unlock the software's full potential. The tips and suggestions presented in this document are aimed at helping readers approach the building of a super-scale or large-scale compute cluster.

Cloud computing is major paradigm shift in IT development with powerful advantages. For any enterprise IT department, standing still is not an option. Public cloud service providers also have a need to continue to evolve and advance their IT infrastructure. key principles of cloud computing infrastructure: efficiency, simplification, security, and open standards.

REFERENCES

1. Hyper-Threading Technology, visit www.intel.com/technology/ platform-technology/hyper-threading/index.htm.
2. QuickPath Technology, visit www.intel.com/technology/ quickpath/index.htm.
3. Intelligent Power Node Manager, visit software.intel.com/sites/Datacentermanager.
4. Data Center Manager, visit software.intel.com/sites/ Datacentermanager
5. SmartFoxServer 2X Performance And Scalability White Paper
6. White Paper: Cloud Computing: Considerations and Next Steps
7. Oracle White Paper—Extreme Scalability Using Oracle Grid Engine Software: Managing Extreme Workloads.
8. Anant Agarwal, Ricardo Bianchini, David Chaiken, Kirk L. Johnson, David Kranz, John Kubiatowicz, Beng- Hong Lim, Kenneth Mackenzie, and Donald Yeung. The MlT Alewife machine: Architecture and Performance. In Proceedirigs of the 22nd Annual

Inrernariorzal Symposium on Computer Archirecture, pages 13, June 1995.

9. Tony Brewer and Greg Astfalk. The evolution of the HP/Convex Exemplar. In Proceedings of COMPCON Spring '97: Forty-Second IEEE Computer Society International Conference, pages 8 1-86, February 1997.
10. L. Censier and P. Feautrier. A New Solution to Coherence Problems in Multicache Systems. IEEE Transaction on Computers C-27(12):1112-1118, Dec. 1978.
11. Asgeir Th. Eiriksson, John Keen, Alex Silbey, Swami Venkataraman, and Michael Woodacre. Origin System Design Methodology and Experience: l M-gate ASlCs and Beyond. In Proceedings of COMPCON Spring **'97:** Forty-Second IEEE Computer Society International Conference, pages 157-164, February 1997.
12. Mike Galles. Scalable Pipelined Interconnect for Distributed Endpoint Routing: The SGI SPIDER chip.