SYLLABUS: SYSTEMS MODELING, CLUSTERING AND VIRTUALIZATION

Scalable Computing over the Internet, Technologies for Network based systems, System models for Distributed and Cloud Computing, Software environments for distributed systems and clouds, Performance, Security And Energy Efficiency

SCALABLE COMPUTING OVER THE INTERNET

- Over the past 60 years, computing technology has undergone a series of platform and environment changes.
- we assess evolutionary changes in machine architecture, operating system platform, network connectivity, and application workload.
- Instead of using a centralized computer to solve computational problems, a parallel and distributed computing system uses multiple computers to solve large-scale problems over the Internet. Thus, distributed computing becomes data-intensive and network-centric.

The Age of Internet Computing:

- Billions of people use the Internet every day. As a result, supercomputer sites and large data centers must provide high-performance computing services to huge numbers of Internet users concurrently.
- Because of this high demand, the Linpack Benchmark for high-performance computing (HPC) applications is no longer optimal for measuring system performance.
- The emergence of computing clouds instead demands high-throughput computing (HTC) systems built with parallel and distributed computing technologies.
- We have to upgrade data centers using fast servers, storage systems, and high-bandwidth networks.

1. The Platform Evolution:

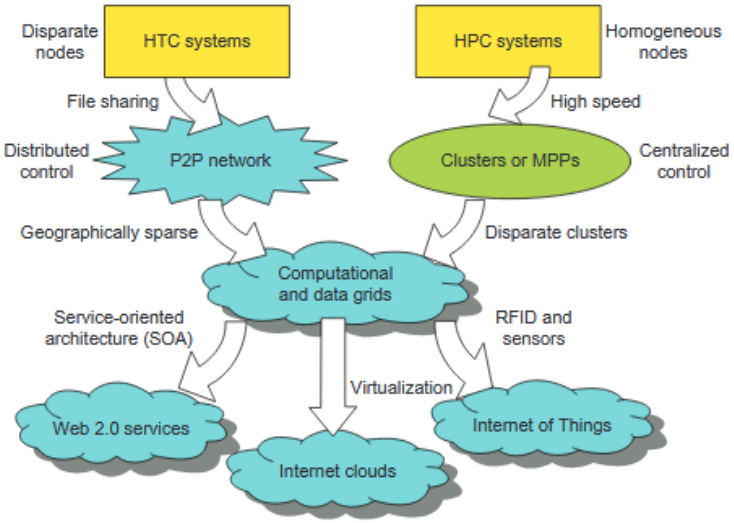
Computer technology has gone through five generations of development, with each generation lasting from 10 to 20 years.

- From 1950 to 1970, a handful of mainframes, including the IBM 360 and CDC 6400, were built to satisfy the demands of large businesses and government organizations.
- From 1960 to 1980, lower-cost minicomputers such as the DEC PDP 11 and VAX Series became popular among small businesses and on college campuses.
- From 1970 to 1990, we saw widespread use of personal computers built with VLSI microprocessors.

- From 1980 to 2000, massive numbers of portable computers and pervasive devices appeared in both wired and wireless applications.
- Since 1990, the use of both HPC and HTC systems hidden in clusters, grids, or Internet clouds has proliferated. These systems are employed by both consumers and high-end web-scale computing and information services.

2.HPC:

- On the HPC side, supercomputers (massively parallel processors or MPPs) are gradually replaced by clusters of cooperative computers out of a desire to share computing resources.
- The cluster is often a collection of homogeneous compute nodes that are physically connected in close range to one another.
- For many years, HPC systems emphasize the raw speed performance. The speed of HPC systems has increased from Gflops in the early 1990s to now Pflops in 2010. This improvement was driven mainly by the demands from scientific, engineering, and manufacturing communities.



3.HTC:

- On the HTC side, peer-to-peer (P2P) networks are formed for distributed file sharing and content delivery applications.
- A P2P system is built over many client machines. Peer machines are globally distributed in nature. P2P, cloud computing, and web service platforms are more focused on HTC applications than on HPC applications.
- This HTC paradigm pays more attention to high-flux computing.

- The main application for high-flux computing is in Internet searches and web services by millions or more users simultaneously.
- The performance goal thus shifts to measure high throughput or the number of tasks completed per unit of time.

Clustering and P2P technologies lead to the development of computational grids or data grids.

4.Three New Computing Paradigms:

1. With the introduction of SOA, Web 2.0 services become available.
2. Advances in virtualization make it possible to see the growth of Internet clouds as a new computing paradigm.
3. The maturity of radio-frequency identification (RFID), Global Positioning System (GPS), and sensor technologies has triggered the development of the Internet of Things (IoT).

5.Computing Paradigm Distinctions:

- The high-technology community has argued for many years about the precise definitions of centralized computing, parallel computing, distributed computing, and cloud computing.
- In general, distributed computing is the opposite of centralized computing.
- The field of parallel computing overlaps with distributed computing to a great extent.
- Cloud computing overlaps with distributed, centralized, and parallel computing.

1. **Centralized computing:** This is a computing paradigm by which all computer resources are centralized in one physical system. All resources (processors, memory, and storage) are fully shared and tightly coupled within one integrated OS. Many data centers and supercomputers are centralized systems, but they are used in parallel, distributed, and cloud computing applications.
2. **Parallel computing:** In parallel computing, all processors are either tightly coupled with centralized shared memory or loosely coupled with distributed memory.
3. **Distributed computing:** This is a field of computer science/engineering that studies distributed systems. A distributed system consists of multiple autonomous computers, each having its own private memory, communicating through a computer network. Information exchange in a distributed system is accomplished through message passing.
4. **Cloud computing:** An Internet cloud of resources can be either a centralized or a distributed computing system. The cloud applies parallel or distributed computing, or both. Clouds can be built with physical or virtualized resources over large data centers that are centralized or distributed.

6.Distributed System Families:

- In the future, both HPC and HTC systems will demand multi-core or many-core processors that can handle large numbers of computing threads per core.
- Both HPC and HTC systems emphasize parallelism and distributed computing. Future HPC and HTC systems must be able to satisfy this huge demand in computing power in terms of throughput, efficiency, scalability, and reliability.
- The system efficiency is decided by speed, programming, and energy factors (i.e., throughput per watt of energy consumed).
- Meeting these goals requires to yield the following design objectives:
  1) Efficiency measures the utilization rate of resources in an execution model by exploiting massive parallelism in HPC. For HTC, efficiency is more closely related to job throughput, data access, storage, and power efficiency.
  2) Dependability measures the reliability and self-management from the chip to the system and application levels. The purpose is to provide high-throughput service with Quality of Service (QoS) assurance, even under failure conditions.
  3) Adaptation in the programming model measures the ability to support billions of job requests over massive data sets and virtualized cloud resources under various workload and service models.
  4) Flexibility in application deployment measures the ability of distributed systems to run well in both HPC (science and engineering) and HTC (business) applications.

SCALABLE COMPUTING TRENDS & NEW PARADIGMS

Several predictable trends in technology are known to drive computing applications. According to Moore's law indicates that processor speed doubles every 18 months.

Gilder's law indicates that network bandwidth has doubled each year in the past. This has also driven the adoption and use of commodity technologies in large-scale computing.

1. Degrees of Parallelism:

- Bit-level parallelism (BLP): converts bit-serial processing to word-level processing gradually.
- Instruction-level parallelism (ILP): Over the years, users graduated from 4-bit microprocessors to 8-, 16-, 32-, and 64-bit CPUs. This led us to the next wave of improvement, known as instruction-level parallelism (ILP), in which the processor executes multiple instructions simultaneously rather than only one instruction at a time.

- Data-level parallelism (DLP): was made popular through SIMD (single instruction, multiple data) and vector machines using vector or array types of instructions. DLP requires even more hardware support and compiler assistance to work properly.
- Task-level parallelism (TLP): A modern processor explores all of the aforementioned parallelism types. However, TLP is far from being very successful due to difficulty in programming and compilation of code for efficient execution on multicore CMPs.
- Job-level parallelism (JLP): As we move from parallel processing to distributed processing, we will see an increase in computing granularity to job-level parallelism (JLP). It is fair to say that coarse-grain parallelism is built on top of fine-grain parallelism.

2. Innovative applications:

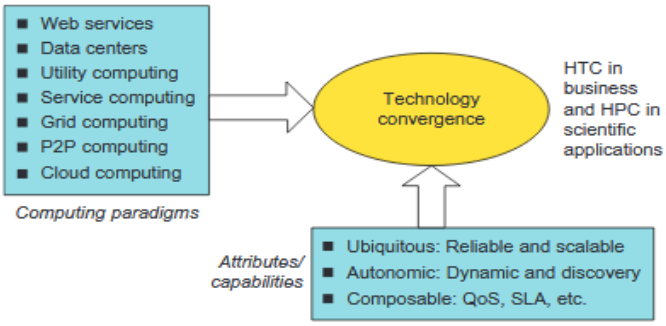
Both HPC and HTC systems desire transparency in many application aspects. For example, data access, resource allocation, process location, concurrency in execution, job replication, and failure recovery should be made transparent to both users and system management. The following table highlights a few key applications that have driven the development of parallel and distributed systems over the years.

Table 1.1 Applications of High-Performance and High-Throughput Systems

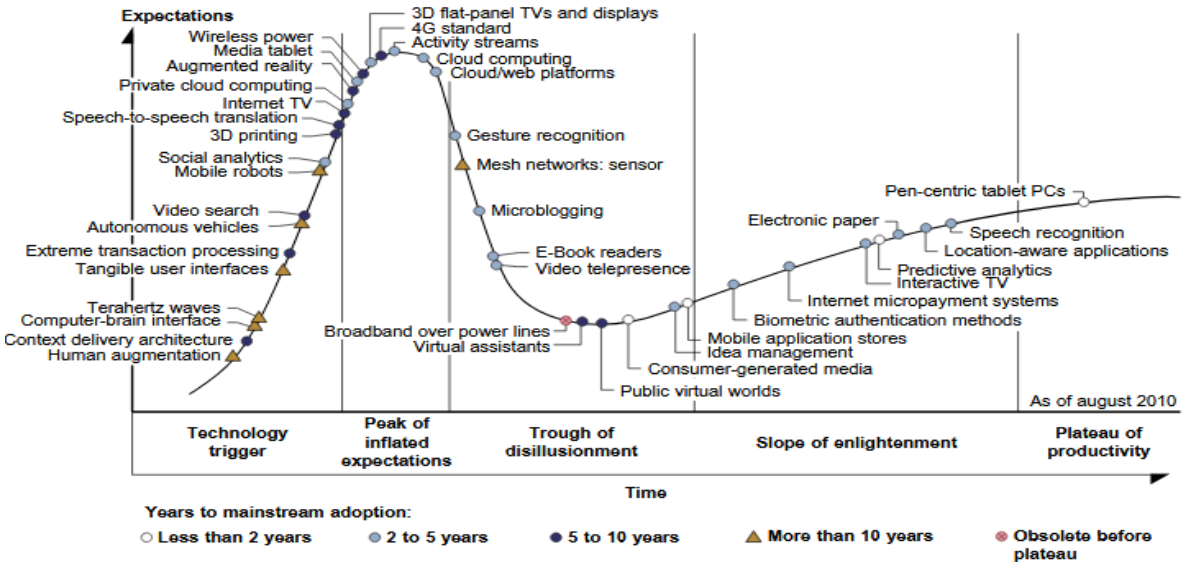
Domain	Specific Applications
Science and engineering	Scientific simulations, genomic analysis, etc. Earthquake prediction, global warming, weather forecasting, etc.
Business, education, services industry, and health care	Telecommunication, content delivery, e-commerce, etc. Banking, stock exchanges, transaction processing, etc. Air traffic control, electric power grids, distance education, etc. Health care, hospital automation, telemedicine, etc.
Internet and web services, and government applications	Internet search, data centers, decision-making systems, etc. Traffic monitoring, worm containment, cyber security, etc. Digital government, online tax return processing, social networking, etc.
Mission-critical applications	Military command and control, intelligent systems, crisis management, etc.

3. The trend towards utility computing:

Utility computing focuses on a business model in which customers receive computing resources from a paid service provider. All grid/cloud platforms are regarded as utility service providers. Cloud computing offers a broader concept than utility computing. Distributed cloud applications run on any available servers in some edge networks.



4. The Hype Cycle of New Technologies : Any new and emerging computing and information technology may go through a hype cycle, as illustrated in Figure. This cycle shows the expectations for the technology at five different stages. The expectations rise sharply from the trigger period to a high peak of inflated expectations. Through a short period of disillusionment, the expectation may drop to a valley and then increase steadily over a long enlightenment period to a plateau of productivity. The number of years for an emerging technology to reach a certain stage is marked by special symbols. The hollow circles indicate technologies that will reach mainstream adoption in two years. The gray circles represent technologies that will reach mainstream adoption in two to five years. The solid circles represent those that require five to 10 years to reach mainstream adoption, and the triangles denote those that require more than 10 years. The crossed circles represent technologies that will become obsolete before they reach the plateau. The cloud technology had just crossed the peak of the expectation stage in 2010, and it was expected to take two to five more years to reach the productivity stage.



5. The Internet of Things:

- The traditional Internet connects machines to machines or web pages to web pages.
- The IoT refers to the networked interconnection of everyday objects, tools, devices, or computers. One can view the IoT as a wireless network of sensors that interconnect all things in our daily life.
- These things can be large or small and they vary with respect to time and place. The idea is to tag every object using RFID or a related sensor or electronic technology such as GPS.
- This communication can be made between people and things or among the things themselves.
- Three communication patterns co-exist: namely
  1) H2H (human-to-human),
  2) H2T (human-to-thing)
  3) T2T (thing-to-thing)

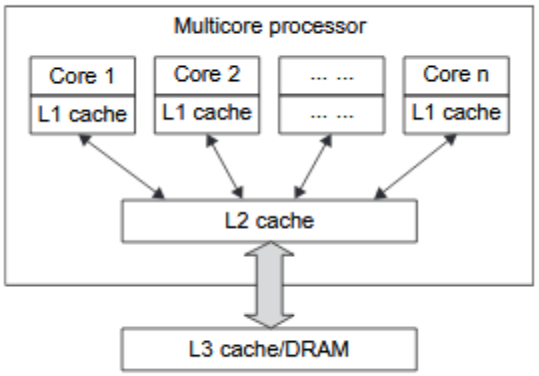
6. Cyber-Physical Systems: A cyber-physical system(CPS) is the result of interaction between computational processes and the physical world. A CPS integrates "cyber"(heterogeneous, asynchronous) with "physical"(concur-rent and information-dense) objects. A CPS merges the"3C"technologies of computation, communication, and control into an intelligent closed feedback system between the physical world and the information world, a concept which is actively explored in the United States. The IoT emphasizes various networking connections among physical objects, while the CPS emphasizes exploration of virtual reality(VR) applications in the physical world. We may transform how we interact with the physical world just like the Internet transformed how we interact with the virtual world.

TECHNOLOGIES FOR NETWORK-BASED SYSTEMS

In particular, we will focus on viable approaches to building distributed operating systems for handling massive parallelism in a distributed environment.

1.Multi core CPUs and Multithreading Technologies:

Both multi-core CPU and many-core GPU processors can handle multiple instruction threads at different magnitudes today. Figure shows the architecture of a typical multi core processor. Each core is essentially a processor with its own private cache (L1 cache). Multiple cores are housed in the same chip with an L2 cache that is shared by all cores. In the future, multiple CMPs could be built on the same CPU chip with even the L3 cache on the chip. Multi core and multi- threaded CPUs are equipped with many high-end processors, including the Intel i7, Xeon, AMD Opteron, Sun Niagara, IBM Power 6, and X cell processors.



Multi core processor

Multi core CPU and Many-Core GPU Architectures:

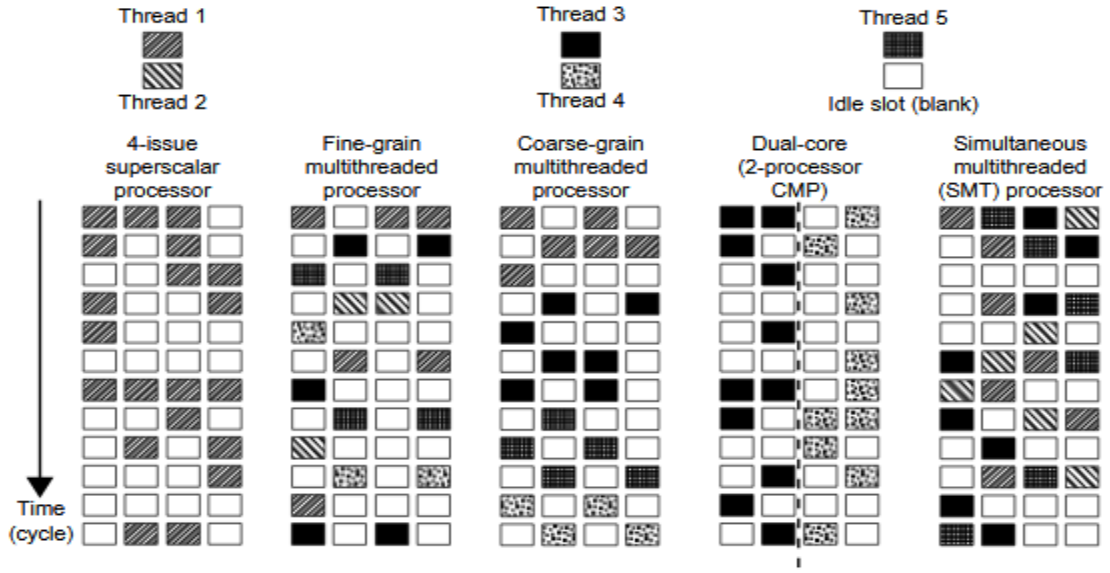
Multi core CPUs may increase from the tens of cores to hundreds or more in the future. But the CPU has reached its limit in terms of exploiting massive DLP due to the aforementioned memory wall problem.

This has triggered the development of many-core GPUs with hundreds or more thin cores.

Multithreading Technology:

1. four-issue superscalar processor
2. a fine-grain multithreaded processor
3. a coarse-grain multithreaded processor
4. a two-core CMP multithreaded (SMT) processor
5. a simultaneous multithreaded (SMT) processor

- The superscalar processor is single-threaded with four functional units. Each of the three multithreaded processors is four-way multithreaded over four functional data paths.
- In the dual-core processor, assume two processing cores, each a single-threaded two-way superscalar processor.

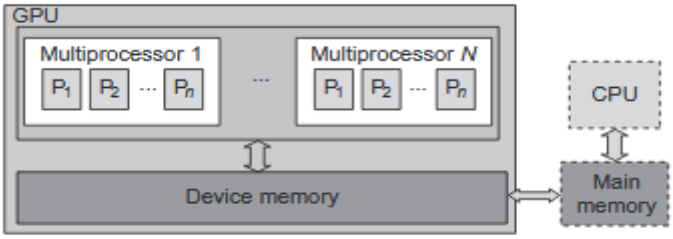


Multi threading models

1) Only instructions from the same thread are executed in a superscalar processor.
2) Fine-grain multithreading switches the execution of instructions from different threads per cycle.
3) Course-grain multithreading executes many instructions from the same thread for quite a few cycles before switching to another thread.
4) The multi core CMP executes instructions from different threads completely.
5) The SMT allows simultaneous scheduling of instructions from different threads in the same cycle.

2. GPU Computing to Exascale and Beyond:

- A GPU is a graphics coprocessor or accelerator mounted on a computer's graphics card or video card. A GPU offloads the CPU from tedious graphics tasks in video editing applications.
- The world's first GPU, the GeForce 256, was marketed by NVIDIA in 1999.
- How GPUs Work:
  - Early GPUs functioned as coprocessors attached to the CPU. Today, the NVIDIA GPU has been upgraded to 128 cores on a single chip.



GPU

- Furthermore, each core on a GPU can handle eight threads of instructions. This translates to having up to 1,024 threads executed concurrently on a single GPU.
- The GPU is optimized to deliver much higher throughput with explicit management of on-chip memory.

3.Memory, Storage, and Wide-Area Networking:

Memory Technology:

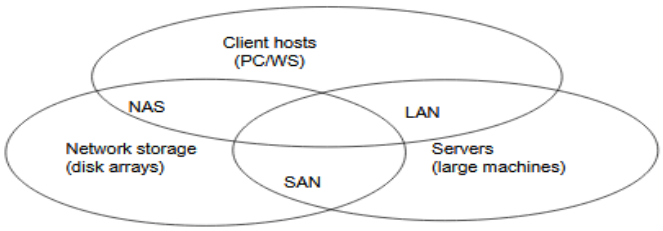
- The growth of DRAM chip capacity from 16 KB in 1976 to 64 GB in 2011. This shows that memory chips have experienced a 4x increase in capacity every three years.
- For hard drives, capacity increased from 260 MB in 1981 to 250 GB in 2004. The Seagate Barracuda XT hard drive reached 3 TB in 2011. This represents an approximately 10x increase in capacity every eight years.
- Faster processor speed and larger memory capacity result in a wider gap between processors and memory. The memory wall may become even worse a problem limiting the CPU performance in the future.

Disks and Storage Technology:

- The rapid growth of flash memory and solid-state drives (SSDs) also impacts the future of HPC and HTC systems.
- A typical SSD can handle 300,000 to 1 million write cycles per block. So the SSD can last for several years, even under conditions of heavy write usage. Flash and SSD will demonstrate impressive speedups in many applications.

System-Area Interconnects:

- The nodes in small clusters are mostly interconnected by an Ethernet switch or a local area network (LAN).
- A LAN typically is used to connect client hosts to big servers.



- A storage area network (SAN) connects servers to network storage such as disk arrays.
- Network attached storage (NAS) connects client hosts directly to the disk arrays.

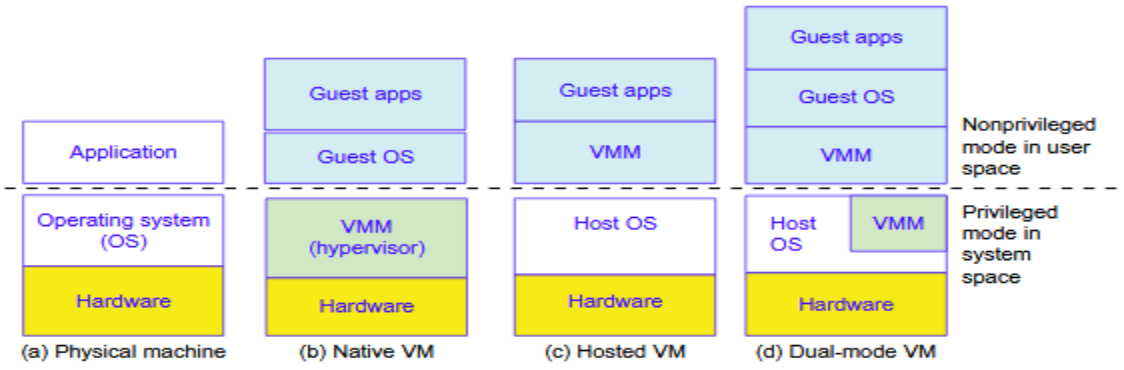
Wide-Area Networking:

- The rapid growth of Ethernet bandwidth from 10 Mbps in 1979 to 1 Gbps in 1999, and 40 ~ 100 GE in 2011. It has been speculated that 1 Tbps network links will become available by 2013.
- An increase factor of two per year on network performance was reported, which is faster than Moore's law on CPU speed doubling every 18 months. The implication is that more computers will be used concurrently in the future.
- High-bandwidth networking increases the capability of building massively distributed systems.

4. Virtual Machines and Virtualization Middleware:

- Virtual machines (VMs) offer novel solutions to underutilized resources, application inflexibility, software manageability, and security concerns in existing physical machines.

1. Virtual Machines:



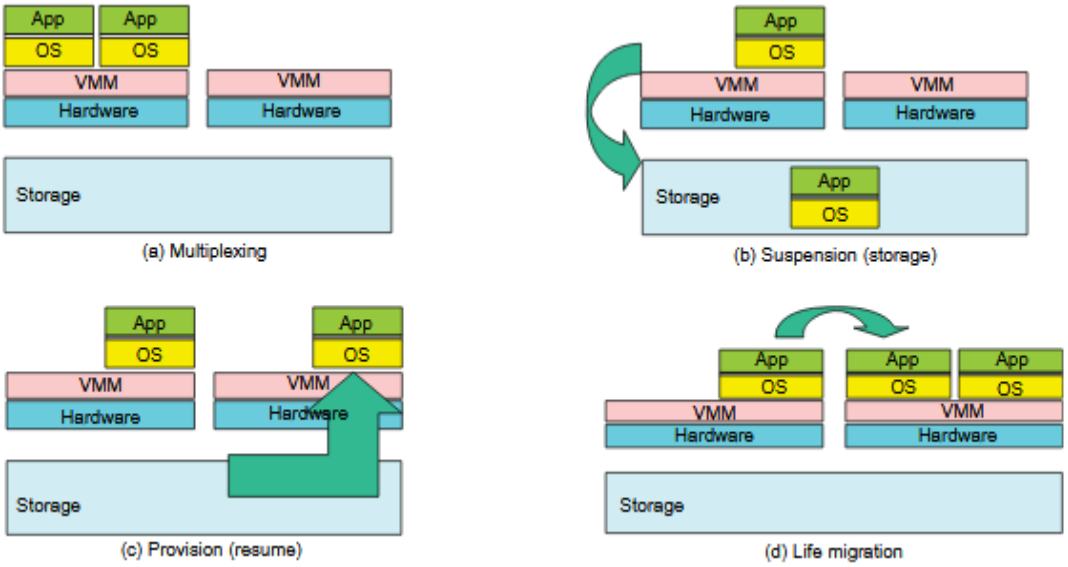
- The host machine is equipped with the physical hardware, as shown at the bottom of the figure. An example is an x-86 architecture desktop running its installed Windows OS, as shown in part (a) of the figure.
- The VM can be provisioned for any hardware system. The VM is built with virtual resources managed by a guest OS to run a specific application. Between the VMs and

the host platform, one needs to deploy a middleware layer called a virtual machine monitor (VMM).

- Figure (b) shows a native VM installed with the use of a VMM called a hypervisor in privileged mode. For example, the hardware has x-86 architecture running the Windows system.
- The guest OS could be a Linux system and the hypervisor is the XEN system developed at Cambridge University. This hypervisor approach is also called bare-metal VM, because the hypervisor handles the bare hardware (CPU, memory, and I/O) directly.
- Another architecture is the host VM shown in Figure (c). Here the VMM runs in non-privileged mode. The host OS need not be modified. The VM can also be implemented with a dual mode, as shown in Figure (d).
- Part of the VMM runs at the user level and another part runs at the supervisor level. In this case, the host OS may have to be modified to some extent. Multiple VMs can be ported to a given hardware system to support the virtualization process.
- The VM approach offers hardware independence of the OS and applications. The user application running on its dedicated OS could be bundled together as a virtual appliance that can be ported to any hardware platform.
- The VM could run on an OS different from that of the host computer.

2. VM Primitive Operations:

1) Multiplexing : VMs can be multiplexed between hardware machines, as shown in Figure (a).
2) Suspension : VM can be suspended and stored in stable storage, as shown in Figure (b).
3) Provision : a suspended VM can be resumed or provisioned to a new hardware platform, as shown in Figure (c).
4) Life migration : VM can be migrated from one hardware platform to another, as shown in Figure(d)



3. Virtual Infrastructure:

Dynamic mapping of system resources to specific applications. The result is decreased costs and increased efficiency and responsiveness. Virtualization for server consolidation and containment is a good example of this

5.Data Center Virtualization for Cloud Computing:

- Cloud architecture is built with commodity hardware and network devices.
- Almost all cloud platforms choose the popular x86 processors. Low-cost terabyte disks and Gigabit Ethernet are used to build data centers.
- Data center design emphasizes storage and energy efficiency are more important than shear speed performance.

1. Data Center Growth and Cost Breakdown:

- A large data center may be built with thousands of servers. Smaller data centers are typically built with hundreds of servers.
- According to a 2009 IDC report typically only 30 percent of data center costs goes toward purchasing IT equipment (such as servers and disks),
- 33 percent is attributed to the chiller (cooling system)
- 18 percent to the uninterruptible power supply (UPS),
- 9 percent to computer room air conditioning (CRAC),
- And the remaining 7 percent to power distribution, lighting, and transformer costs.
- Thus, about 60 percent of the cost to run a data center is allocated to management and maintenance.

2. Low cost design philosophy : High-end switches or routers may be too cost-prohibitive for building data centers. Thus, using high-bandwidth networks may not fit the economics of cloud computing. Commodity switches and networks are more desirable in data centers. Similarly, using commodityx86 servers is more desired over expensive mainframes. The software layer handles network traffic balancing, fault tolerance, and expandability. Currently, nearly all cloud computing data centers use Ethernet as their fundamental network technology

3. Convergence of Technologies: Essentially, cloud computing is enabled by the convergence of technologies in four areas: (1) hard-ware virtualization and multi-core chips, (2) utility and grid computing, (3) SOA, Web 2.0, and WS mashups, and (4) atonomic computing and data center automation

SYSTEM MODELS FOR DISTRIBUTED AND CLOUD COMPUTING

Distributed and cloud computing systems are built over a large number of autonomous computer nodes. These node machines are interconnected by SANs, LANs, or WANs in a hierarchical manner.

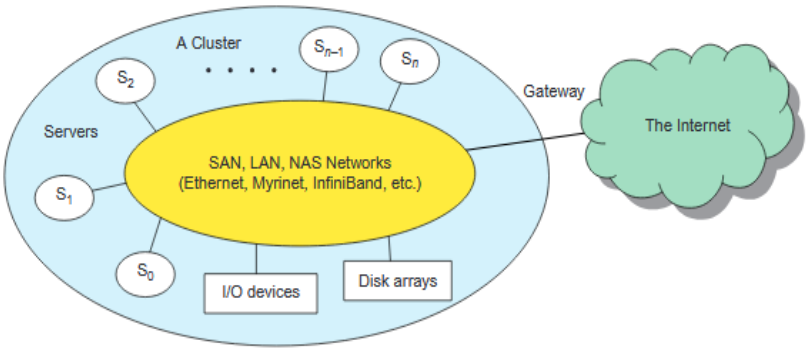
- Massive systems are classified into four groups:
  1. Clusters
  2. P2P networks
  3. Computing grids,
  4. Internet clouds over huge data centers.

Table 1.2 Classification of Parallel and Distributed Computing Systems

Functionality, Applications	Computer Clusters [10,28,38]	Peer-to-Peer Networks [34,46]	Data/ Computational Grids [6,18,51]	Cloud Platforms [1,9,11,12,30]
Architecture, Network Connectivity, and Size	Network of compute nodes interconnected by SAN, LAN, or WAN hierarchically	Flexible network of client machines logically connected by an overlay network	Heterogeneous clusters interconnected by high-speed network links over selected resource sites	Virtualized cluster of servers over data centers via SLA
Control and Resources Management	Homogeneous nodes with distributed control, running UNIX or Linux	Autonomous client nodes, free in and out, with self-organization	Centralized control, server-oriented with authenticated security	Dynamic resource provisioning of servers, storage, and networks
Applications and Network-centric Services	High-performance computing, search engines, and web services, etc.	Most appealing to business file sharing, content delivery, and social networking	Distributed supercomputing, global problem solving, and data center services	Upgraded web search, utility computing, and outsourced computing services
Representative Operational Systems	Google search engine, SunBlade, IBM Road Runner, Cray XT4, etc.	Gnutella, eMule, BitTorrent, Napster, KaZaA, Skype, JXTA	TeraGrid, GriPhyN, UK EGEE, D-Grid, ChinaGrid, etc.	Google App Engine, IBM Bluecloud, AWS, and Microsoft Azure

1. Cluster Architecture: An interconnected stand alone computers that works co-operatively as a single integrated computing resource is called a cluster

- The architecture of a typical server cluster built around a low-latency, high-bandwidth interconnection network.
- To build a larger cluster with more nodes, the interconnection network can be built with multiple levels of Gigabit Ethernet, Myrinet, or InfiniBand switches.
- Through hierarchical construction using a SAN, LAN, or WAN, one can build scalable clusters with an increasing number of nodes.
- The cluster is connected to the Internet via a virtual private network (VPN) gateway. The gateway IP address locates the cluster.
- The system image of a computer is decided by the way the OS manages the shared cluster resources.
- Most clusters have loosely coupled node computers. All resources of a server node are managed by their own OS.
- Thus, most clusters have multiple system images as a result of having many autonomous nodes under different OS control.



Cluster of servers interconnected by high bandwidth SAN or LAN

Single-System Image:

- Cluster should merge multiple system images into a single-system image (SSI).
- Cluster designers desire a cluster operating system or some middleware to support SSI at various levels, including the sharing of CPUs, memory, and I/O across all cluster nodes.

- An SSI is an illusion created by software or hardware that presents a collection of resources as one integrated, powerful resource.

Hardware, Software, and Middleware Support

- Clusters exploring massive parallelism are commonly known as MPPs. Almost all HPC clusters in the Top 500 list are also MPPs. The building blocks are computer nodes (PCs, workstations, servers, or
- SMP), special communication software such as PVM or MPI, and a network interface card in each
- computer node. Most clusters run under the Linux OS.
- The computer nodes are interconnected by a high-bandwidth network (such as Gigabit Ethernet, Myrinet, InfiniBand, etc.).
- Special cluster middleware supports are needed to create SSI or high availability (HA). Both
- sequential and parallel applications can run on the cluster, and special parallel environments are
- needed to facilitate use of the cluster resources.

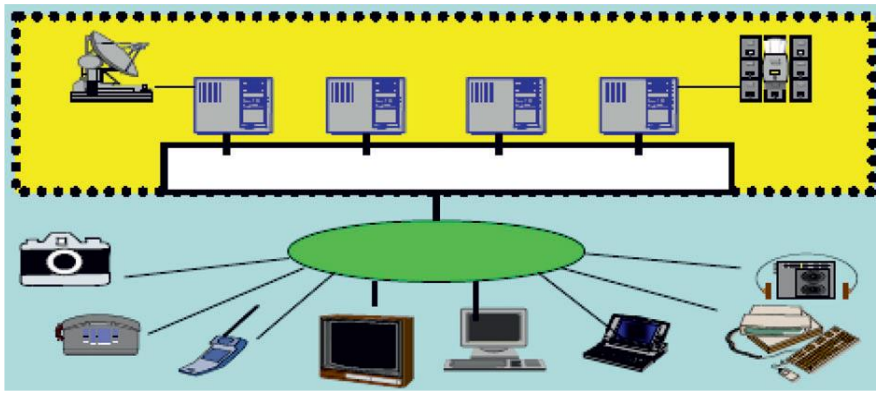
Major Cluster Design Issues

A cluster-wide OS for complete resource sharing is not available yet. Middleware or OS extensions were developed at the user space to achieve SSI at selected functional levels. Without this middleware, cluster nodes cannot work together effectively to achieve cooperative computing. The software environments and applications must rely on the middleware to achieve high performance. The cluster benefits come from scalable performance, efficient message passing, high system availability, seamless fault tolerance, and cluster-wide job management,

2. Grid Computing Infrastructures:

Grid families:

- In grid systems are classified in essentially two categories: computational or data grids and P2P grids.



Computing grid

Table 1.4 Two Grid Computing Infrastructures and Representative Systems

Design Issues	Computational and Data Grids	P2P Grids
Grid Applications Reported	Distributed supercomputing, National Grid initiatives, etc.	Open grid with P2P flexibility, all resources from client machines
Representative Systems	TeraGrid built in US, ChinaGrid in China, and the e-Science grid built in UK	JXTA, FightAid@home, SETI@home
Development Lessons Learned	Restricted user groups, middleware bugs, protocols to acquire resources	Unreliable user-contributed resources, limited to a few apps

Computational Grids:

- Like an electric utility power grid, a computing grid offers an infrastructure that couples computers, software/middleware, special instruments, and people and sensors together. The grid is often constructed across LAN, WAN, or Internet backbone networks at a regional, national, or global scale.
- The resource sites offer complementary computing resources, including workstations, large servers, a mesh of processors, and Linux clusters to satisfy a chain of computational needs.

3. Peer-to-Peer Network Families:

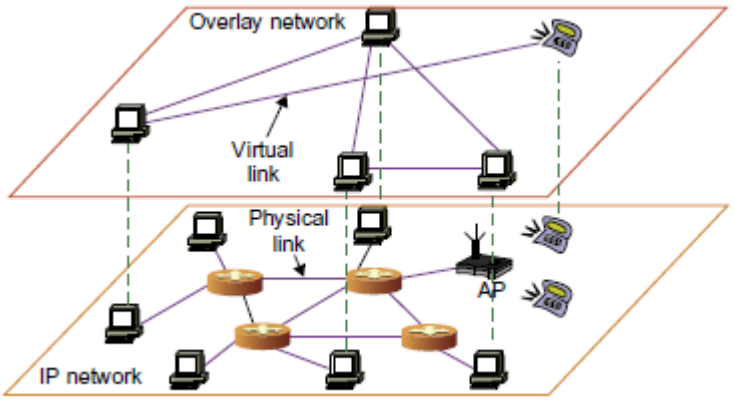
- client machines (PCs and workstations) are connected to a central server for compute, e-mail, file access, and database applications.
- The P2P architecture offers a distributed model of networked systems. First, a P2P network is client-oriented instead of server-oriented.
- P2P systems are introduced at the physical level and overlay networks at the logical level.

P2P Systems:

- In a P2P system, every node acts as both a client and a server, providing part of the system resources.
- Peer machines are simply client computers connected to the Internet. All client machines act autonomously to join or leave the system freely.
- This implies that no master-slave relationship exists among the peers. No central coordination or central database is needed.
- The system is self-organizing with distributed control.
- Only the participating peers form the physical network at any time.
- Unlike the cluster or grid, a P2P network does not use a dedicated interconnection network. The physical network is simply an ad hoc network formed at various Internet domains randomly using the TCP/IP and NAI protocols.

Overlay Networks:

- Data items or files are distributed in the participating peers.
- Based on communication or file-sharing needs, the peer IDs form an overlay network at the logical level. This overlay is a virtual network formed by mapping each physical machine with its ID, logically, through a virtual mapping.



Overlay networks built with virtual links

P2P Application Families:

- Based on application, P2P networks are classified into four groups.
  1) Distributed File Sharing
  2) Collaboration Platform

3) Distributed P2P Computing
4) P2P Platform

P2P Computing Challenges:

- P2P computing faces three types of heterogeneity problems in hardware, software, and network requirements.
- Different network connections and protocols make it too complex to apply in real applications.
- System scaling is directly related to performance and bandwidth. P2P networks do have these properties. Data locality, network proximity, and interoperability are three design objectives in distributed P2P applications.
- Fault tolerance, failure management, and load balancing are other important issues in using overlay network.
- Security, privacy, and copyright violations are major worries by those in the industry in terms of applying P2P technology in business applications
- By replicating data in multiple peers, one can easily lose data in failed nodes.

Disadvantages of P2P networks : Because the system is not centralized, managing it is difficult.

- In addition, the system lacks security. Anyone can log on to the system and cause damage or abuse.
- Further, all client computers connected to a P2P network cannot be considered reliable or virus-free. P2P networks are reliable for a small number of peer nodes. They are only useful for applications that require a low level of security and have no concern for data sensitivity.

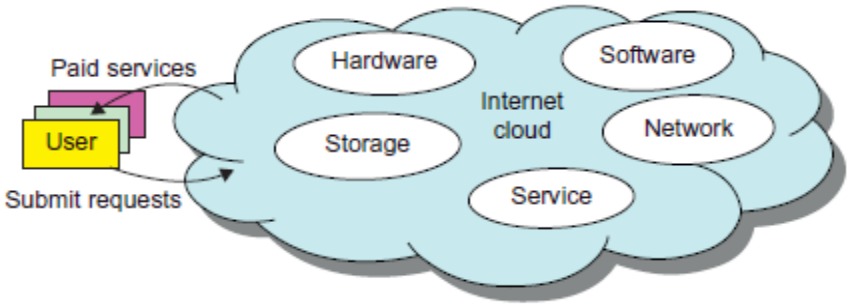
Cloud Computing over the Internet:

- In the future, working with large data sets will typically mean sending the computations (programs) to the data, rather than copying the data to the workstations.
- This reflects the trend in IT of moving computing and data from desktops to large data centers where there is on-demand provision of software, hardware, and data as a service.
- This data explosion has promoted the idea of cloud computing.
- Based on this definition, a cloud allows workloads to be deployed and scaled out quickly through rapid provisioning of virtual or physical machines.
- The cloud supports redundant, self-recovering, highly scalable programming models that allow workloads to recover from many unavoidable hardware/software failures.

- Finally, the cloud system should be able to monitor resource use in real time to enable rebalancing of allocations when needed.

Internet Clouds:

- Cloud computing applies a virtualized platform with elastic resources on demand by provisioning hardware, software, and data sets dynamically (see Figure). The idea is to move desktop computing to a service-oriented platform using server clusters and huge databases at data centers.



Virtualized resources from data centers to form a cloud

The Cloud Landscape:

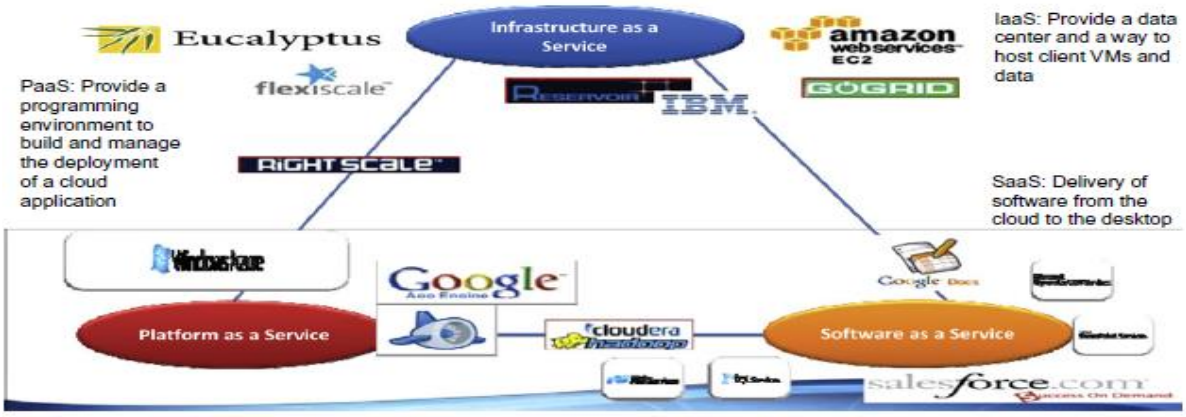
- These traditional systems have encountered several performance bottlenecks: constant system maintenance, poor utilization, and increasing costs associated with hardware/software upgrades.
- Cloud computing as an on-demand computing paradigm resolves or relieves us from these problems.
- The cloud landscape and major cloud players, based on three cloud service models.
    1. Infrastructure as a Service (IaaS)
    2. Platform as a Service (PaaS)
    3. Software as a Service (SaaS)
- **Infrastructure as a Service (IaaS):** This model puts together infrastructures demanded by users—namely servers, storage, networks, and the data center fabric. The user can deploy and run on multiple VMs running guest OSes on specific applications. The user does not manage or control the underlying cloud infrastructure, but can specify when to request and release the needed resources.

- **Platform as a Service (PaaS):** This model enables the user to deploy user-built applications onto a virtualized cloud platform. PaaS includes middleware, databases, development tools, and some runtime support such as Web 2.0 and Java. The platform includes both hardware and software integrated with specific programming interfaces. The provider supplies the API and software tools (e.g., Java, Python, Web 2.0, .NET). The user is freed from managing the cloud infrastructure.
- **Software as a Service (SaaS):** This refers to browser-initiated application software over thousands of paid cloud customers. The SaaS model applies to business processes, industry applications, consumer relationship management (CRM), enterprise resources planning (ERP), human resources (HR), and collaborative applications. On the customer side, there is no upfront investment in servers or software licensing. On the provider side, costs are rather low, compared with conventional hosting of user applications.

The following list highlights eight reasons to adapt the cloud for upgraded Internet applications and web services:

Desired location in areas with protected space and higher energy efficiency

1) Sharing of peak-load capacity among a large pool of users, improving overall utilization
2) Separation of infrastructure maintenance duties from domain-specific application development
3) Significant reduction in cloud computing cost, compared with traditional computing paradigms
4) Cloud computing programming and application development
5) Service and data discovery and content/service distribution
6) Privacy, security, copyright, and reliability issues
7) Service agreements, business models, and pricing policies



SOFTWARE ENVIRONMENTS FOR DISTRIBUTED SYSTEMS AND CLOUDS

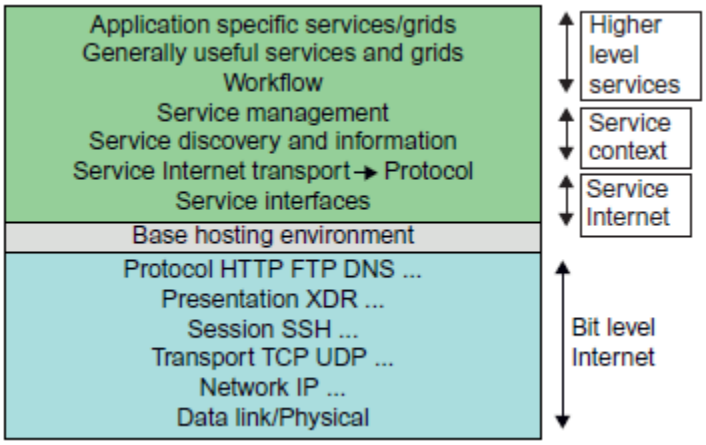
Service-Oriented Architecture (SOA):

- In grids/web services, Java, and CORBA, an entity is, respectively, a service, a Java object, and a CORBA distributed object in a variety of languages.
- These architectures build on the traditional seven Open Systems Interconnection (OSI) layers that provide the base networking abstractions.
- On top of this we have a base software environment, which would be .NET or Apache Axis for web services, the Java Virtual Machine for Java, and a broker network for CORBA.
- On top of this base environment one would build a higher level environment reflecting the special features of the distributed computing environment.
- This starts with entity interfaces and inter-entity communication, which rebuild the top four OSI layers but at the entity and not the bit level.

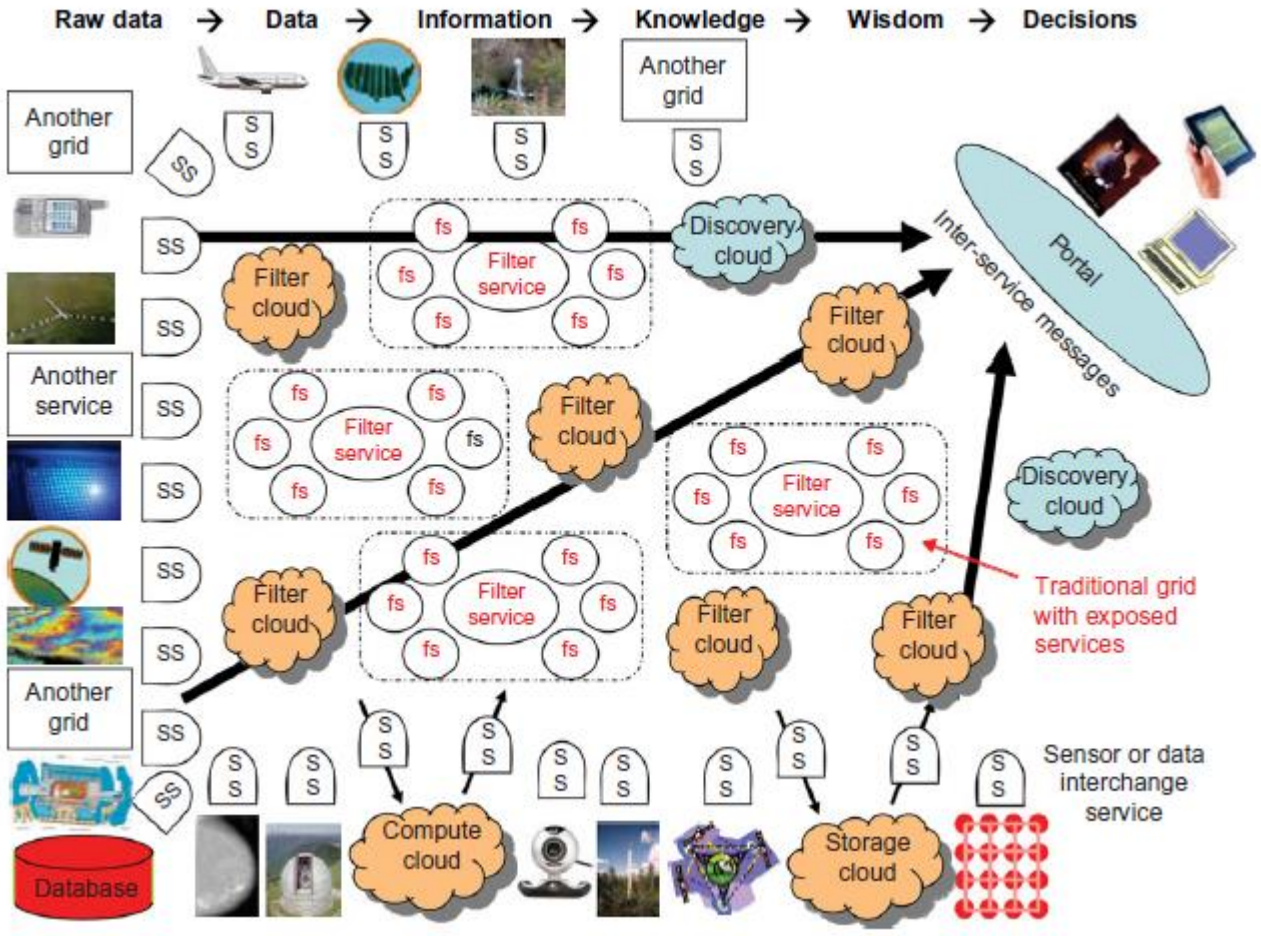
Layered Architecture for Web Services and Grids:

1) Higher Level services
2) Service Context
3) Service Internet
4) Bit-Level Internet

- The entity interfaces correspond to the Web Services Description Language (WSDL).
- Java method, and CORBA interface definition language (IDL) specifications in these example distributed systems.
- These interfaces are linked with customized, high-level communication systems: SOAP, RMI, and IIOP in the three examples.
- These communication systems support features including particular message patterns (such as Remote Procedure Call or RPC), fault recovery, and specialized routing.
- Often, these communication systems are built on message-oriented middleware (enterprise bus) infrastructure such as WebSphere MQ or Java Message Service (JMS),
- which provide rich functionality and support virtualization of routing, senders, and recipients.



The Evolution of SOA:



- SOA applies to building grids, clouds, grids of clouds, clouds of grids, clouds of clouds (also known as inter-clouds), and systems of systems in general.
- Raw data → Data → Information → Knowledge → Wisdow → Decisions
- A large number of sensors provide data-collection services, denoted in the figure as SS.
- A sensor can be a ZigBee device, a Bluetooth device, a WiFi access point, a personal computer, a GPA, or a wireless phone, among other things.
- All the SS devices interact with large or small computers, many forms of grids, databases, the compute cloud, the storage cloud, the filter cloud, the discovery cloud, and so on.
- Filter services (fs in the figure) are used to eliminate unwanted raw data, in order to respond to specific requests from the web, the grid, or web services.
- SOA aims to search for, or sort out, the useful data from the massive amounts of raw data items.
- Processing this data will generate useful information, and subsequently, the knowledge for our daily use.
- Finally, we make intelligent decisions based on both biological and machine wisdom.

Trends toward Distributed Operating Systems:

- The computers in most distributed systems are loosely coupled. Thus, a distributed system inherently has multiple system images.
- This is mainly due to the fact that all node machines run with an independent operating system.
- To promote resource sharing and fast communication among node machines, it is best to have a distributed OS that manages all resources coherently and efficiently.

Distributed Operating Systems:

- Tanenbaum identifies three approaches for distributing resource management functions in a distributed computer system.
  1) The first approach is to build a network OS over a large number of heterogeneous OS platforms.
  2) The second approach is to develop middleware to offer a limited degree of resource sharing, similar to the MOSIX/OS developed for clustered systems.
  3) The third approach is to develop a truly distributed OS to achieve higher use or system transparency.

Amoeba versus DCE:

- DCE is a middleware-based system for distributed computing environments.
- The Amoeba was academically developed at Free University in the Netherlands.
- To balance the resource management workload, the functionalities of such a distributed OS should be distributed to any available server. to any available server. In this sense, the conventional OS runs only on a centralized platform.
- With the distribution of OS services, the distributed OS design should take a lightweight microkernel approach like the Amoeba or should extend an existing OS like the DCE by extending UNIX.
- The trend is to free users from most resource management duties.

MOSIX2 for Linux Clusters:

- MOSIX2 is a distributed OS, which runs with a virtualization layer in the Linux environment.
- This layer provides a partial single-system image to user applications.
- MOSIX2 supports both sequential and parallel applications, and discovers resources and migrates software processes among Linux nodes.
- Flexible management of a grid allows owners of clusters to share their computational resources among multiple cluster owners.

Transparency in Programming Environments:

- The user data, applications, OS, and hardware are separated into four levels.
- Data is owned by users, independent of the applications.
- The OS provides clear interfaces, standard programming interfaces, or system calls to application programmers.
- In future cloud infrastructure, the hardware will be separated by standard interfaces from the OS.
- Thus, users will be able to choose from different OSes on top of the hardware devices they prefer to use.
- To separate user data from specific application programs, users can enable cloud applications as SaaS.
- Thus, users can switch among different services.

Parallel and Distributed Programming Models:

1) **Message-Passing Interface (MPI):**
   - This is the primary programming standard used to develop parallel and concurrent programs to run on a distributed system.

- MPI is essentially a library of subprograms that can be called from C or FORTRAN to write parallel programs running on a distributed system.
- The idea is to embody clusters, grid systems, and P2P systems with upgraded web services and utility computing applications.

2) Map Reduce:

- This is a web programming model for scalable data processing on large clusters over large data sets.
- The model is applied mainly in web-scale search and cloud computing applications.
- The user specifies a Map function to generate a set of intermediate key/value pairs.
- Then the user applies a Reduce function to merge all intermediate values with the same intermediate key.
- Map Reduce is highly scalable to explore high degrees of parallelism at different job levels.
- A typical Map Reduce computation process can handle terabytes of data on tens of thousands or more client machines.

3) Hadoop Library:

- The package enables users to write and run applications over vast amounts of distributed data.
- Users can easily scale Hadoop to store and process petabytes of data in the web space.
- It is efficient, as it processes data with a high degree of parallelism across a large number of commodity nodes.
- it is reliable in that it automatically keeps multiple data copies to facilitate redeployment of computing tasks upon unexpected system failures.

4) Open Grid Services Architecture:

- OGSA is a common standard for general public use of grid services.
- Genesis II is a realization of OGSA. Key features include a distributed execution environment,
- Public Key Infrastructure (PKI) services using a local certificate authority (CA),
- Trust management, and security policies in grid computing.

5) Globus Toolkits and Extensions:

- Globus is a middleware library.
- This library implements some of the OGSA standards for resource discovery, allocation, and security enforcement in a grid environment.
- The Globus packages support multisite mutual authentication with PKI certificates.
- In addition, IBM has extended Globus for business applications.

PERFORMANCE, SECURITY, AND ENERGY EFFICIENCY

1. Performance Metrics:

- In a distributed system, performance is attributed to a large number of factors.
- System throughput is often measured in MIPS, Tflops (tera floating-point operations per second), or TPS (transactions per second).
- Other measures include job response time and network latency. An interconnection network that has low latency and high bandwidth is preferred.
- System overhead is often attributed to OS boot time, compile time, I/O data rate, and the runtime support system used.
- Other performance-related metrics include the QoS for Internet and web services.
- System availability and dependability.
- Security resilience for system defense against network attacks.

Dimensions of Scalability:

1) **Size scalability:**
   - This refers to achieving higher performance or more functionality by increasing the machine size.
   - The word "size" refers to adding processors, cache, memory, storage, or I/O channels.
   - The most obvious way to determine size scalability is to simply count the number of processors installed.
2) **Software scalability:**
   - This refers to upgrades in the OS or compilers, adding mathematical and engineering libraries, porting new application software, and installing more user-friendly programming environments.
   - Some software upgrades may not work with large system configurations.
   - Testing and fine-tuning of new software on larger systems is a nontrivial job.
3) **Application scalability:**
   - This refers to matching problem size scalability with machine size scalability.
   - Problem size affects the size of the data set or the workload increase. Instead of increasing machine size, users can enlarge the problem size to enhance system efficiency or cost-effectiveness.

4) Technology scalability:

- This refers to a system that can adapt to changes in building technologies, such as the component and networking technologies.
- When scaling a system design with new technology one must consider three aspects: time, space, and heterogeneity.
  1) Time refers to generation scalability. When changing to new-generation processors, one must consider the impact to the motherboard, power supply, packaging and cooling, and so forth.
  2) Space is related to packaging and energy concerns. Technology scalability demands harmony and portability among suppliers.
  3) Heterogeneity refers to the use of hardware components or software packages from different vendors. Heterogeneity may limit the scalability.

2. Fault Tolerance and System Availability

In addition to performance, system availability and application flexibility are two other important design goals in a distributed computing system.

System Availability:

- HA (high availability) is desired in all clusters, grids, P2P networks, and cloud systems. A system is highly available if it has a long mean time to failure (MTTF) and a short mean time to repair (MTTR).
- System availability is formally defined as follows:

  System Availability =MTTF/ðMTTF +MTTRÞ (1.5)

- System availability is attributed to many factors. All hardware, software, and network components may fail. Any failure that will pull down the operation of the entire system is called a single point of failure. In general, as a distributed system increases in size, availability decreases due to a higher chance of failure and a difficulty in isolating the failures.
- Both SMP and MPP are very vulnerable with centralized resources under one OS. NUMA machines have improved in availability due to the use of multiple OSes.
- Most clusters are designed to have HA with failover capability. Meanwhile, private clouds are created out of virtualized data centers
- A grid is visualized as a hierarchical cluster of clusters. Grids have higher availability due to the isolation of faults. Therefore, clusters, clouds, and grids have decreasing availability as the system increases in size. A P2P file-sharing network has the highest aggregation of client machines.

Network Threats and Data Integrity

Clusters, grids, P2P networks, and clouds demand security and copyright protection if they are to be accepted in today's digital society.

1. **Threats to Systems and Networks**

- Network viruses have threatened many users in widespread attacks. These incidents have created a worm epidemic by pulling down many routers and servers, and are responsible for the loss of billions of dollars in business, government, and services.
- Loss of data integrity may be caused by user alteration, Trojan horses, and service spoofing attacks. A denial of service (DoS) results in a loss of system operation and Internet connections. Lack of authentication or authorization leads to attackers' illegitimate use of computing resources. Open resources such as data centers, P2P networks, and grid and cloud infrastructures could become the next targets. Users need to protect clusters, grids, clouds, and P2P systems. Otherwise, users should not use or trust them for outsourced work.

2. **Security Responsibilities**

- Three security requirements are often considered: confidentiality, integrity, and availability for most Internet service providers and cloud users.
- In the order of SaaS, PaaS, and IaaS, the providers gradually release the responsibility of security control to the cloud users. In summary, the SaaS model relies on the cloud provider to perform all security functions.
- At the other extreme, the IaaS model wants the users to assume almost all security functions, but to leave availability in the hands of the providers. The PaaS model relies on the provider to maintain data integrity and availability, but burdens the user with confidentiality and privacy control.
- We also need to deploy mechanisms to prevent online piracy and copyright violations of digital content. Security responsibilities are divided between cloud providers and users differently for the three cloud service models. The providers are totally responsible for platform availability.
- The IaaS users are more responsible for the confidentiality issue. The IaaS providers are more responsible for data integrity. In PaaS and SaaS services, providers and users are equally responsible for preserving data integrity and confidentiality.

3. Copyright Protection

- Collusive piracy is the main source of intellectual property violations within the boundary of a P2P
- network. Paid clients (colluders) may illegally share copyrighted content files with unpaid clients (pirates).
- Online piracy has hindered the use of open P2P networks for commercial content delivery.

  One can develop a proactive content poisoning scheme to stop colluders and pirates from alleged copyright infringements in P2P file sharing.
- Pirates are detected in a timely manner with identity-based signatures and time stamped tokens. This scheme stops collusive piracy from occurring without hurting legitimate P2P clients.

4. System Defense Technologies

- Three generations of network defense technologies have appeared in the past. In the first generation, tools were designed to prevent or avoid intrusions. These tools usually manifested themselves as access control policies or tokens, cryptographic systems, and so forth. However, an intruder could always penetrate a secure system because there is always a weak link in the security provisioning process.
- The second generation detected intrusions in a timely manner to exercise remedial actions. These techniques included firewalls, intrusion detection systems (IDSes), PKI services, reputation systems, and so on.
- The third generation provides more intelligent responses to intrusions.

5. Data Protection Infrastructure

- Security infrastructure is required to safeguard web and cloud services. At the user level, one

  needs to perform trust negotiation and reputation aggregation over all users.
- At the application end, we need to establish security precautions in worm containment and intrusion detection

Energy Efficiency in Distributed Computing:

- Primary performance goals in conventional parallel and distributed computing systems are high performance and high throughput, considering some form of performance reliability (e.g., fault tolerance and security).

- These systems recently encountered new challenging issues including energy efficiency, and workload and resource outsourcing. These emerging issues are crucial not only on their own, but also for the sustainability of large-scale computing systems in general.

- Protection of data centers demands integrated solutions. Energy consumption in parallel and distributed computing systems raises various monetary, environmental, and system performance issues.

Energy Consumption of Unused Servers

- To run a server farm (data center) a company has to spend a huge amount of money for hardware, software, operational support, and energy every year.
- Therefore, companies should thoroughly identify whether their installed server farm (more specifically, the volume of provisioned resources) is at an appropriate level, particularly in terms of utilization.
- It was estimated in the past that, on average, one-sixth (15 percent) of the full-time servers in a company are left powered on without being actively used (i.e., they are idling) on a daily basis. This indicates that with 44 million servers in the world, around 4.7 million servers are not doing any useful work.

Reducing Energy in Active Servers

In addition to identifying unused/underutilized servers for energy savings, it is also necessary to apply appropriate techniques to decrease energy consumption in active distributed systems with negligible influence on their performance. Power management issues in distributed computing platforms can be categorized into four layers: the application layer, middleware layer, resource layer, and network layer.

1. Application Layer:

- Most user applications in science, business, engineering, and financial areas tend to increase a system's speed or quality. By introducing energy-aware applications, the challenge is to design sophisticated multilevel and multi-domain energy management applications without hurting performance.
- The first step toward this end is to explore a relationship between performance and energy consumption. Indeed, an application's energy consumption depends strongly on the number
- of instructions needed to execute the application and the number of transactions with the storage unit (or memory). These two factors (compute and storage) are correlated and they affect completion time.

2. Middleware Layer:

- The middleware layer acts as a bridge between the application layer and the resource layer. This layer provides resource broker, communication service, task analyzer, task scheduler, security access, reliability control, and information service capabilities.
- It is also responsible for applying energy-efficient techniques, particularly in task scheduling. Until recently, scheduling was aimed at minimizing makespan, that is, the execution time of a set of tasks. Distributed computing systems necessitate a new cost function covering both makespan and energy consumption.

3. Resource Layer:

- The resource layer consists of a wide range of resources including computing nodes and storage units. This layer generally interacts with hardware devices and the operating system; therefore, it is responsible for controlling all distributed resources in distributed computing systems.
- Several mechanisms have been developed for more efficient power management of hardware and operating systems. The majority of them are hardware approaches particularly for processors.
- Dynamic power management (DPM) and dynamic voltage-frequency scaling (DVFS) are two popular methods incorporated into recent computer hardware systems.
- In DPM, hardware devices, such as the CPU, have the capability to switch from idle mode to one or more lower power modes.
- In DVFS, energy savings are achieved based on the fact that the power consumption in CMOS circuits has a direct relationship with frequency and the square of the voltage supply. Execution time and power consumption are controllable by switching among different frequencies and voltages.

4. Network Layer

- Routing and transferring packets and enabling network services to the resource layer are the main responsibility of the network layer in distributed computing systems.
- The major challenge to build energy-efficient networks is, again, determining how to measure, predict, and create a balance between energy consumption and performance. Two major challenges to designing energy-efficient networks are:
    - The models should represent the networks comprehensively as they should give a full understanding of interactions among time, space, and energy.
    - New, energy-efficient routing algorithms need to be developed. New, energy-efficient protocols should be developed against network attacks.

- As information resources drive economic and social development, data centers become increasingly important in terms of where the information items are stored and processed, and where services are provided.

- Data centers become another core infrastructure, just like the power grid and transportation systems.