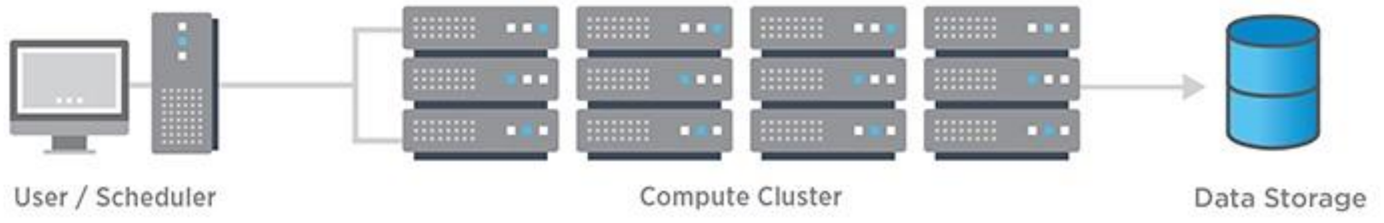
High Performance Computing most generally refers to the practice of aggregating **computing** power in a way that delivers much **higher performance** than one could get out of a typical desktop **computer** or workstation in order to solve large problems in science, engineering, or business.

To build a high-performance computing architecture, compute servers are networked together into a cluster. Software programs and algorithms are run simultaneously on the servers in the cluster. The cluster is networked to the data storage to capture the output. Together, these components operate seamlessly to complete a diverse set of tasks.



The nodes in each cluster work in parallel with each other, boosting processing speed to deliver high-performance computing.

Ex:- Research labs, Media and entertainment, Oil and gas, Artificial intelligence and machine learning, Financial services

High Throughput Computing (HTC) involves running many independent tasks that require a large amount of computing power. With HTC, users can run many copies of their software simultaneously across many different computers for long periods.

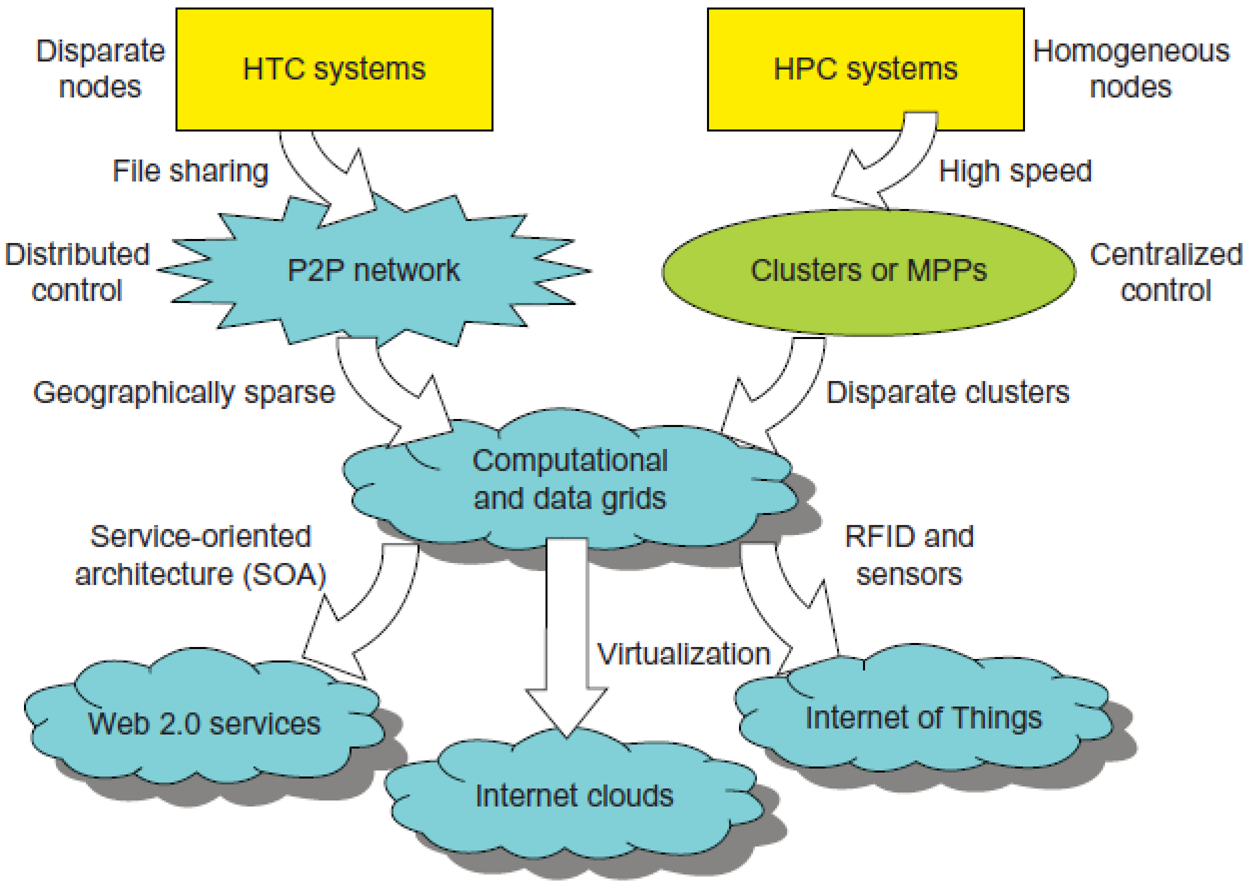
More precisely, it allows many copies of the same program to run *in parallel* or *concurrently*.

Running multiple copies of *exactly* the same problem is obviously a fairly pointless exercise but the power of HTC lies in its ability to use *different data for each program copy* (features in Condor makes this particularly straightforward).

The multiple copies are referred to as *jobs*. Applications might involve jobs processing:

- different patient data in large scale biomedical trials
- different parts of a genome or protein sequence in bioinformatics applications
- different random numbers in a simulations based on Monte Carlo methods
- different model parameters in ensemble simulations or explorations of parameter spaces

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.



- The development of market-oriented high-end computing systems is undergoing a strategic change from an HPC paradigm to an HTC paradigm.
- 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.
- HTC technology needs to not only improve in terms of batch processing speed, but also address the acute problems of cost, energy savings, security, and reliability at many data and enterprise computing centers

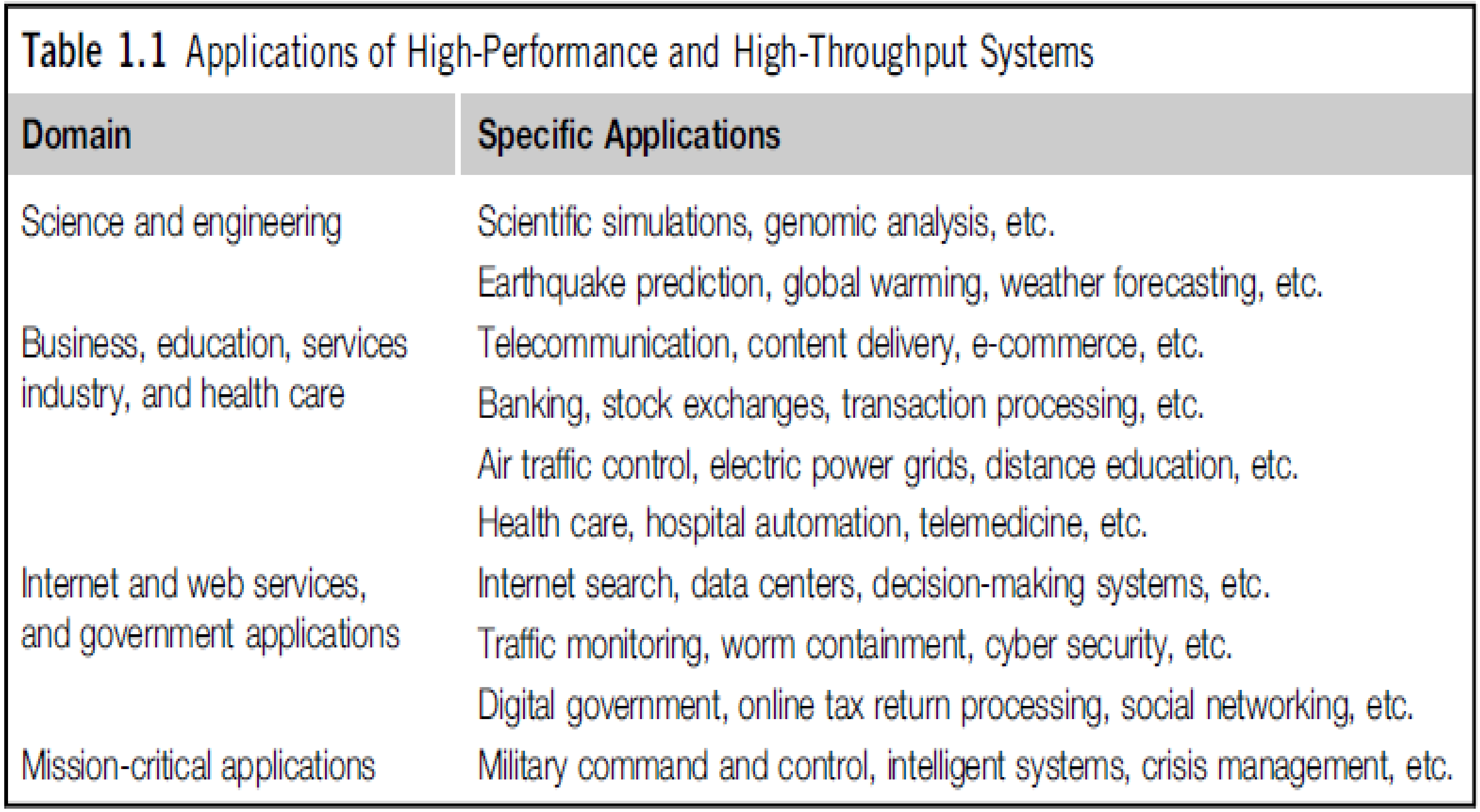


Table 1.1 Applications of High-Performance and High-Throughput Systems

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.

Design objectives of HPC and HTC

In the future, both HPC and HTC systems will demand multicore or many-core processors that can handle large numbers of computing threads per core. Both HPC and HTC systems emphasize parallelism and distributed computing.

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.

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.

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.

Flexibility in application deployment measures the ability of distributed systems to run well in both HPC (science and engineering) and HTC (business) applications.

Computing Paradigms

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

Parallel computing

In this computing, all processors are either tightly coupled with centralized shared memory or loosely coupled with distributed memory.

Interprocessor communication is accomplished through shared memory or via message passing.

A computer system capable of parallel computing is commonly known as a parallel computer [28].
Programs running in a parallel computer are called parallel programs.

The process of writing parallel programs is often referred to as parallel programming

Distributed computing

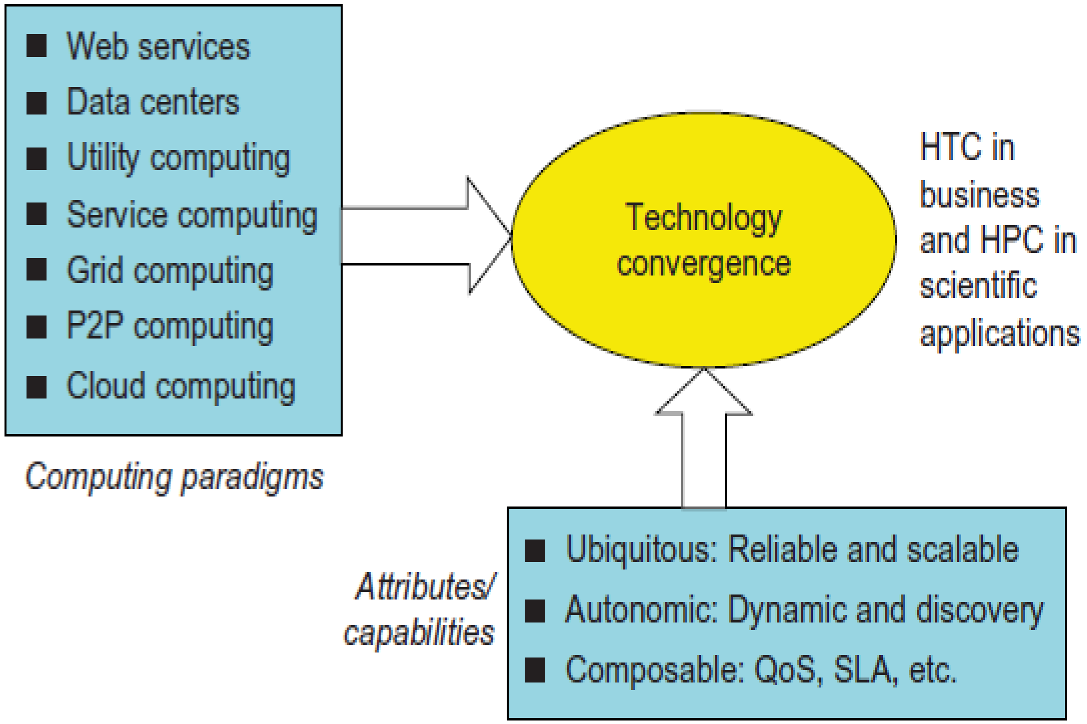
A distributed system consists of multiple autonomous computers, each having its own private memory, communicating through a computer network.

Information exchange in a distributed system is accomplished through message passing.
A computer program that runs in a distributed system is known as a distributed program. The process of writing distributed programs is referred to as distributed programming.

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.

Some authors consider cloud computing to be a form of utility computing or service computing **SOA**.



Web services
Data centers
Utility computing
Service computing
Grid computing
P2P computing
Cloud computing
Computing paradigms
Technology convergence
HTC in business and HPC in scientific applications
Attributes/ capabilities
Ubiquitous: Reliable and scalable
Autonomic: Dynamic and discovery
Composable: QoS, SLA, etc.

TECHNOLOGIES FOR NETWORK-BASED SYSTEMS

1.Multicore CPUs and Multithreading Technologies

The growth of component and network technologies over the past 30 years. They are crucial to the development of HPC and HTC systems.

- processor speed is measured in millions of instructions per second (MIPS)
- network bandwidth is measured in megabits per second (Mbps) or gigabits per second (Gbps). The unit GE refers to 1 Gbps Ethernet bandwidth.

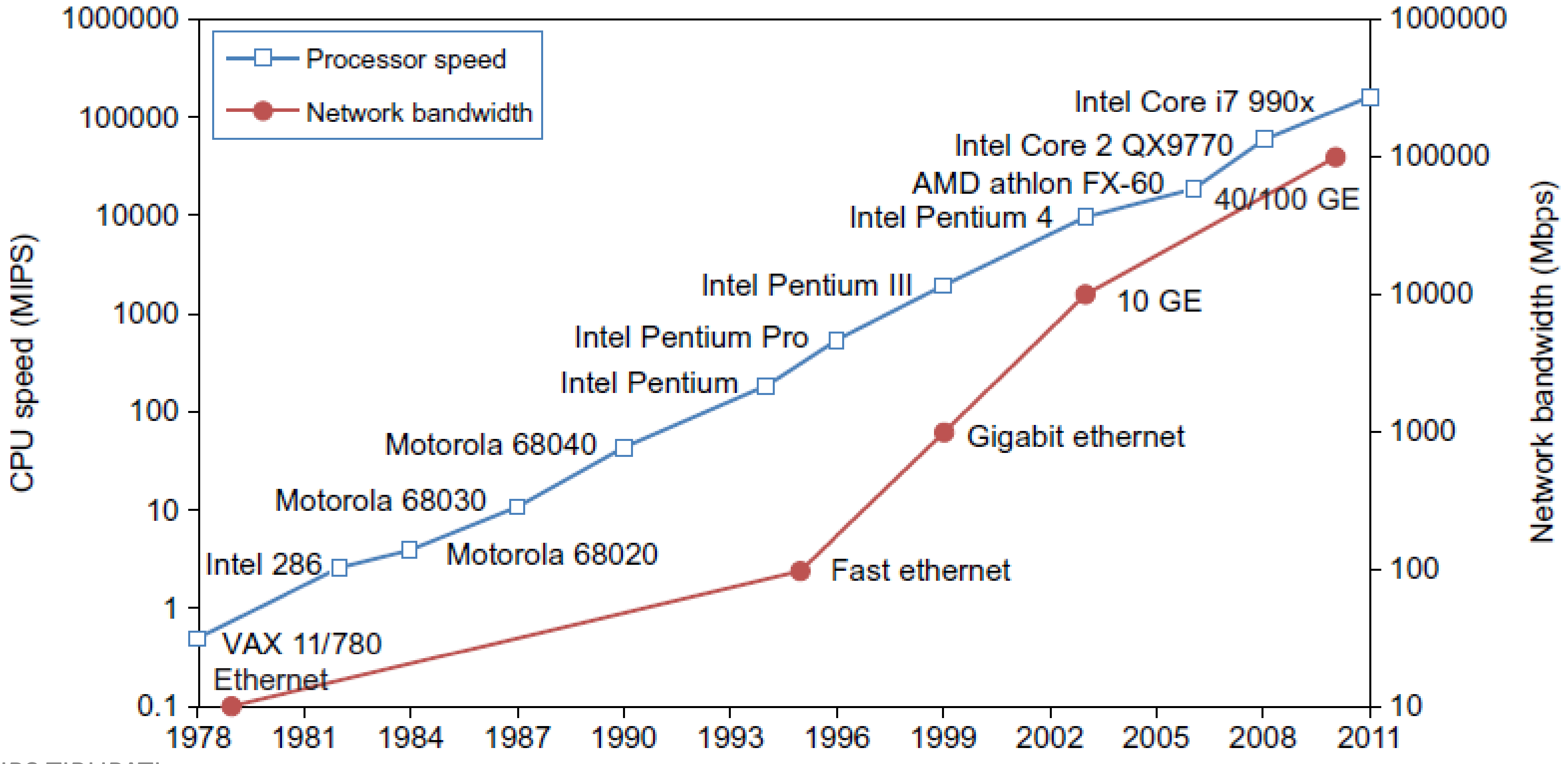
CPU Processors

Today, advanced CPUs or microprocessor chips assume a multicore architecture with dual, quad, six, or more processing cores.

These processors exploit parallelism at ILP and TLP levels

As the figure shows, Moore's law has proven to be pretty accurate in this case. The clock rate for these processors increased from 10 MHz for the Intel 286 to 4 GHz for the Pentium 4 in 30 years.

However, the clock rate reached its limit on CMOS-based chips due to power limitations. At the time of this writing, very few CPU chips run with a clock rate exceeding 5 GHz. In other words, clock rate will not continue to improve unless chip technology matures. This limitation is attributed primarily to excessive heat generation with high frequency or high voltages



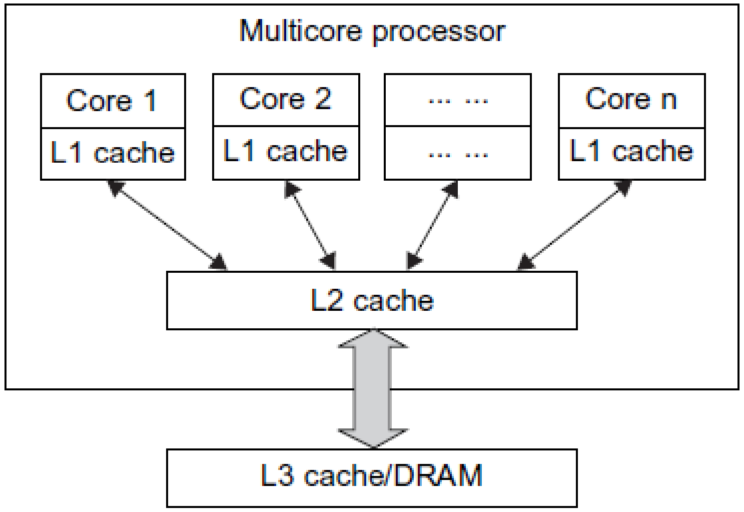
- The ILP is highly exploited in modern CPU processors. ILP mechanisms include multiple-issue superscalar architecture, dynamic branch prediction, and speculative execution, among others.

- These ILP techniques demand hardware and compiler support.

- In addition, DLP and TLP are highly explored in graphics processing units (GPUs) that adopt a many-core architecture with hundreds to thousands of simple cores.

Both multi-core CPU and many-core GPU processors can handle multiple instruction threads at different magnitudes today. Figure 1.5 shows the architecture of a typical multicore processor.

Multicore and multithreaded CPUs are equipped with many high-end processors



Multicore CPU and Many-Core GPU Architectures

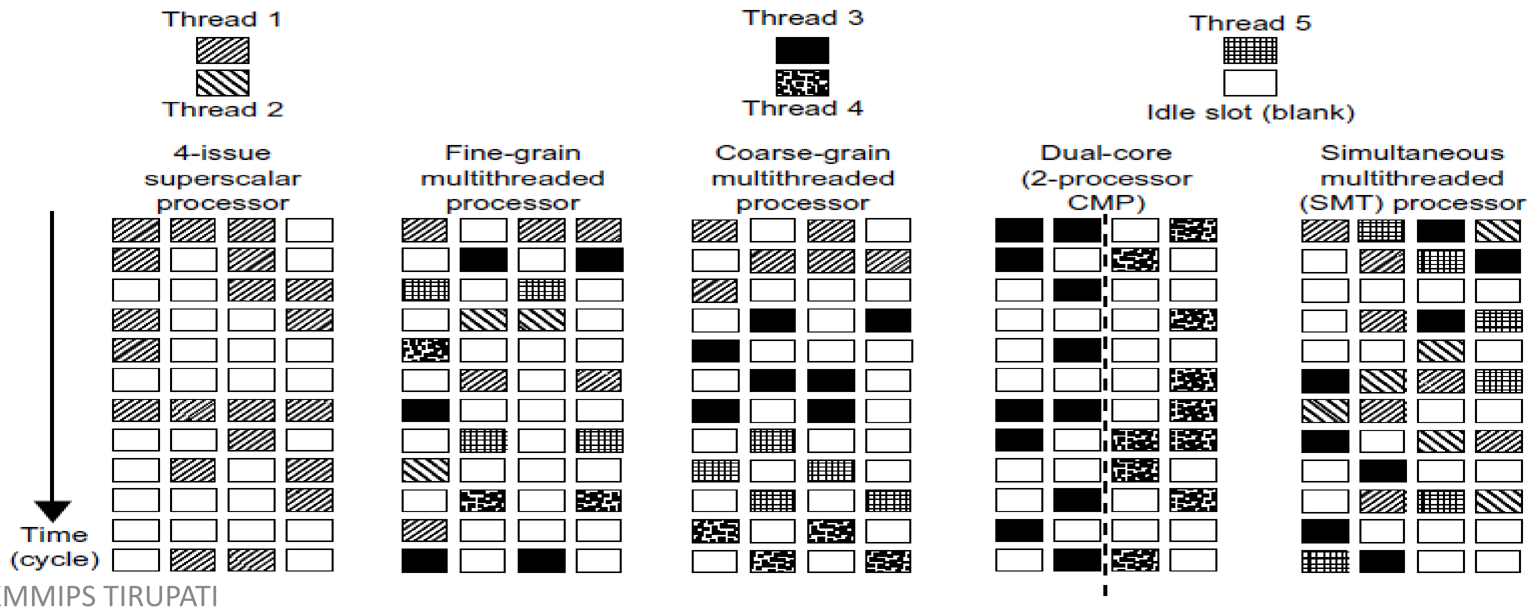
Multicore CPUs may increase from the tens of cores to hundreds or more in the future. But the CPU has reached its limit in terms of exploiting massive DLP due to the aforementioned memory wall problem.

- This has triggered the development of many-core GPUs with hundreds or more thin cores.
- Now, x-86 processors have been extended to serve HPC and HTC systems in some high-end server processors.
- RISC processors have been replaced with multicore x-86 processors and many-core GPUs in the Top 500 systems.
- This trend indicates that x-86 upgrades will dominate in data centers and super computers.
- The GPU also has been applied in large clusters to build supercomputers in MPPs.

Multithreading Technology

the dispatch of five independent threads of instructions to four pipelined data paths (functional units) in each of the following five processor categories, from left to right:

- a four-issue superscalar processor, a fine-grain multithreaded processor, a coarse-grain multithreaded processor, a two-core CMP, and 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.


Thread 1
Thread 2
Thread 3
Thread 4
Thread 5
Idle slot (blank)
4-issue superscalar processor
Fine-grain multithreaded processor
Coarse-grain multithreaded processor
Dual-core (2-processor CMP)
Simultaneous multithreaded (SMT) processor
Time (cycle)

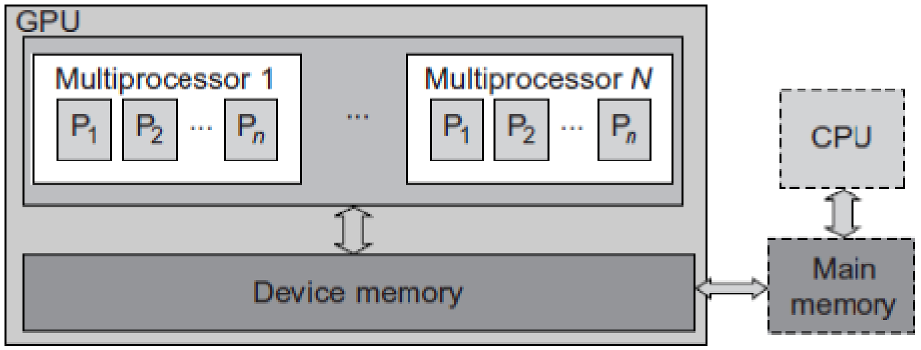
- Only instructions from the same thread are executed in a superscalar processor.
- Fine-grain multithreading switches the execution of instructions from different threads per cycle
- Course-grain multithreading executes many instructions from the same thread for quite a few cycles before switching to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

- The blank squares correspond to no available instructions for an instruction data path at a particular processor cycle.
- More blank cells imply lower scheduling efficiency.
- The maximum ILP or maximum TLP is difficult to achieve at each processor cycle

GPU

- 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.
- These GPU chips can process a minimum of 10 million polygons per second, and are used in nearly every computer on the market today.
- Some GPU features were also integrated into certain CPUs.
- Traditional CPUs are structured with only a few cores.
- However, a modern GPU chip can be built with hundreds of processing cores. Unlike CPUs, GPUs have a throughput architecture that exploits massive parallelism by executing many concurrent threads slowly, instead of executing a single long thread in a conventional microprocessor very quickly.
- Lately, parallel GPUs or GPU clusters have been garnering a lot of attention against the use of CPUs with limited parallelism.
- General-purpose computing on GPUs, known as GPGPUs, have appeared in the HPC field.

- Early GPUs functioned as coprocessors attached to the CPU.
- Today, the NVIDIA GPU has been upgraded to 128 cores on a single chip. 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. This is true massive parallelism, compared to only a few threads that can be handled by a conventional CPU.
- The CPU is optimized for latency caches, while the GPU is optimized to deliver much higher throughput with explicit management of on-chip memory.
- The CPU chip consumes about 2 nJ/instruction, while the GPU chip requires 200 pJ/instruction, which is 1/10 less than that of the CPU.
- The CPU is optimized for latency in caches and memory, while the GPU is optimized for throughput with explicit management of on-chip memory.



In the future, thousand-core GPUs may appear in Exascale (Eflops or 1018 flops) systems. This reflects a trend toward building future MPPs with hybrid architectures of both types of processing chips.

In a DARPA report published in September 2008, four challenges are identified for exascale computing: (1) energy and power, (2) memory and storage, (3) concurrency and locality, and (4) system resiliency.

The CPU chip consumes about 2 nJ/instruction, while the GPU chip requires 200 pJ/instruction, which is 1/10 less than that of the CPU.

The CPU is optimized for latency in caches and memory, while the GPU is optimized for throughput with explicit management of on-chip memory.

Memory Technology

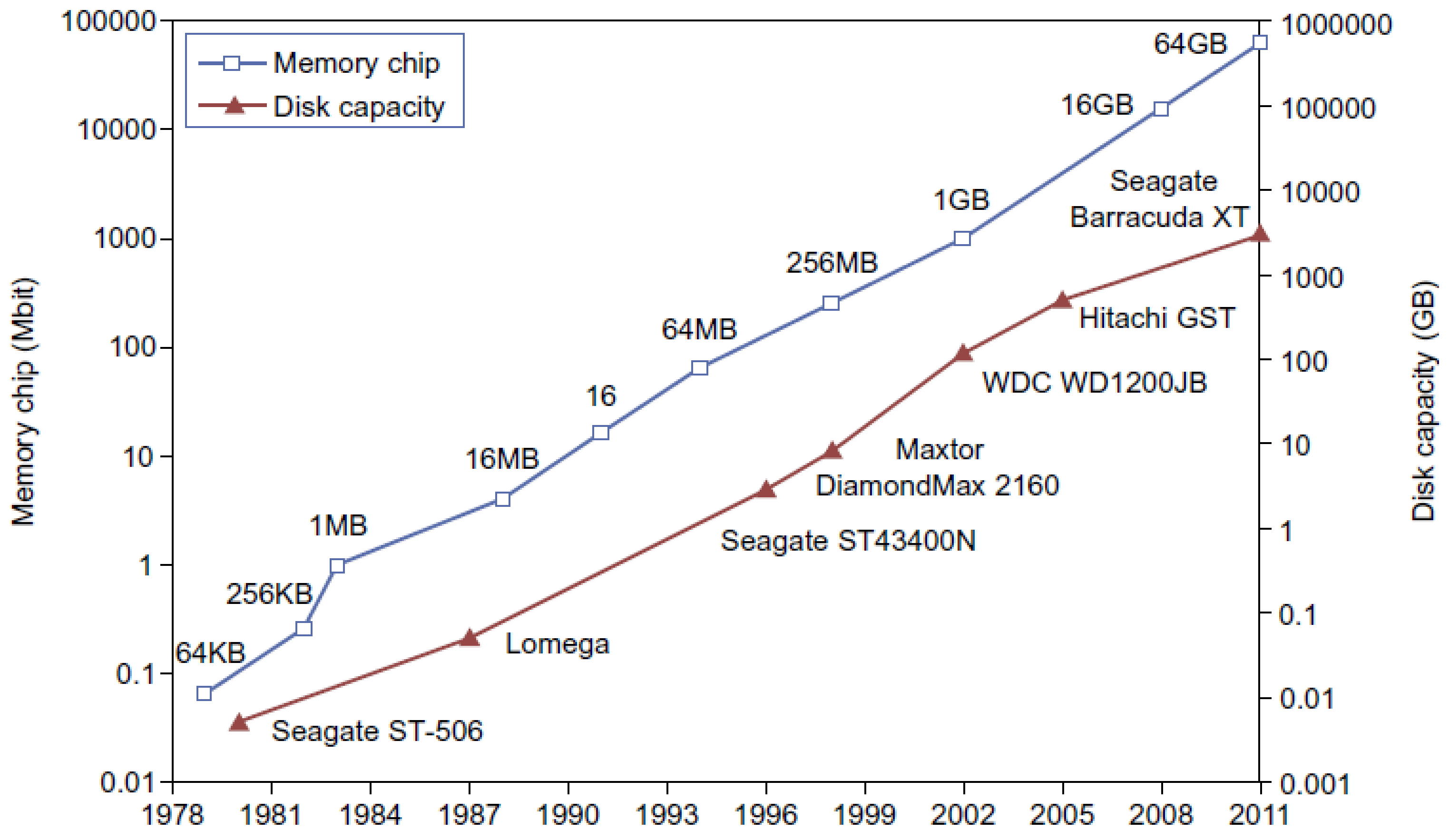
The upper curve in Figure 1.10 plots the growth of DRAM chip capacity from 16 KB in 1976 to 64 GB in 2011.

This shows that memory chips have experienced a 4x increase in capacity every three years. Memory access time did not improve much in the past. In fact, the memory wall problem is getting worse as the processor gets faster.

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.

The capacity increase of disk arrays will be even greater in the years to come. 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.


Memory chip
Disk capacity
Memory chip (Mbit)
Disk capacity (GB)
100000
10000
1000
100
10
1
0.1
0.01
1000000
100000
10000
1000
100
10
1
0.1
0.01
0.001
1978
1981
1984
1987
1990
1993
1996
1999
2002
2005
2008
2011
64KB
256KB
1MB
16MB
16
64MB
256MB
1GB
16GB
64GB
Seagate ST-506
Lomega
Seagate ST43400N
Maxtor DiamondMax 2160
WDC WD1200JB
Hitachi GST
Seagate Barracuda XT

Disks and Storage Technology

Beyond 2011, disks or disk arrays have exceeded 3 TB in capacity. The lower curve in Figure 1.10 shows the disk storage growth in 7 orders of magnitude in 33 years. The rapid growth of flash memory and solid-state drives (SSDs) also impacts the future of HPC and HTC systems.

The mortality rate of SSD is not bad at all. A typical SSD can handle 300,000 to 1 million write cycles per 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.

Eventually, power consumption, cooling, and packaging will limit large system development.

Power increases linearly with respect to clock frequency and quadratic ally with respect to voltage applied on chips.

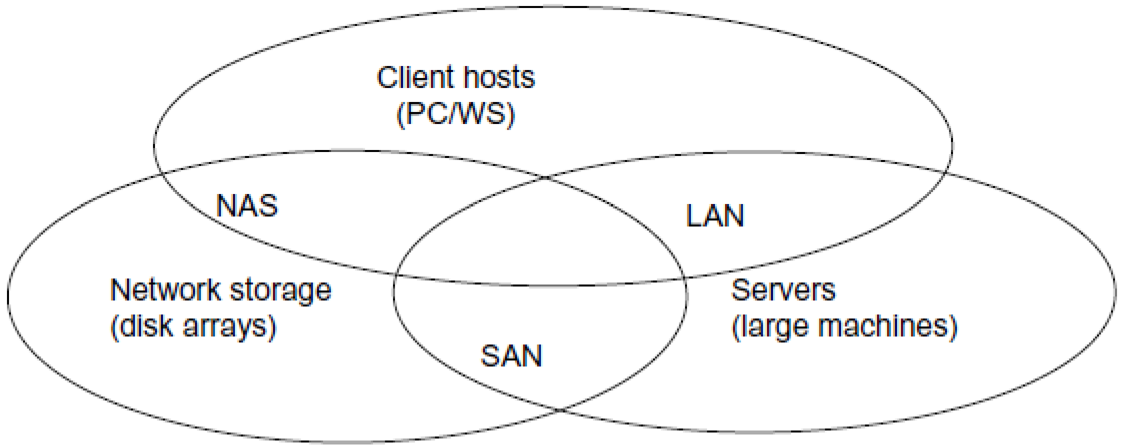
System-Area Interconnects

The nodes in small clusters are mostly interconnected by an Ethernet switch or a local area network (LAN). As Figure 1.11 shows, 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.

All three types of networks often appear in a large cluster built with commercial network components.

If no large distributed storage is shared, a small cluster could be built with a multiport Gigabit Ethernet switch plus copper cables to link the end machines.



Wide area Network

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.

network links with 1,000, 1,000, 100, 10, and 1 Gbps bandwidths were reported, respectively, for international, national, organization, optical desktop, and copper desktop connections in 2006.

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.

Virtual Machines and Virtualization Middleware

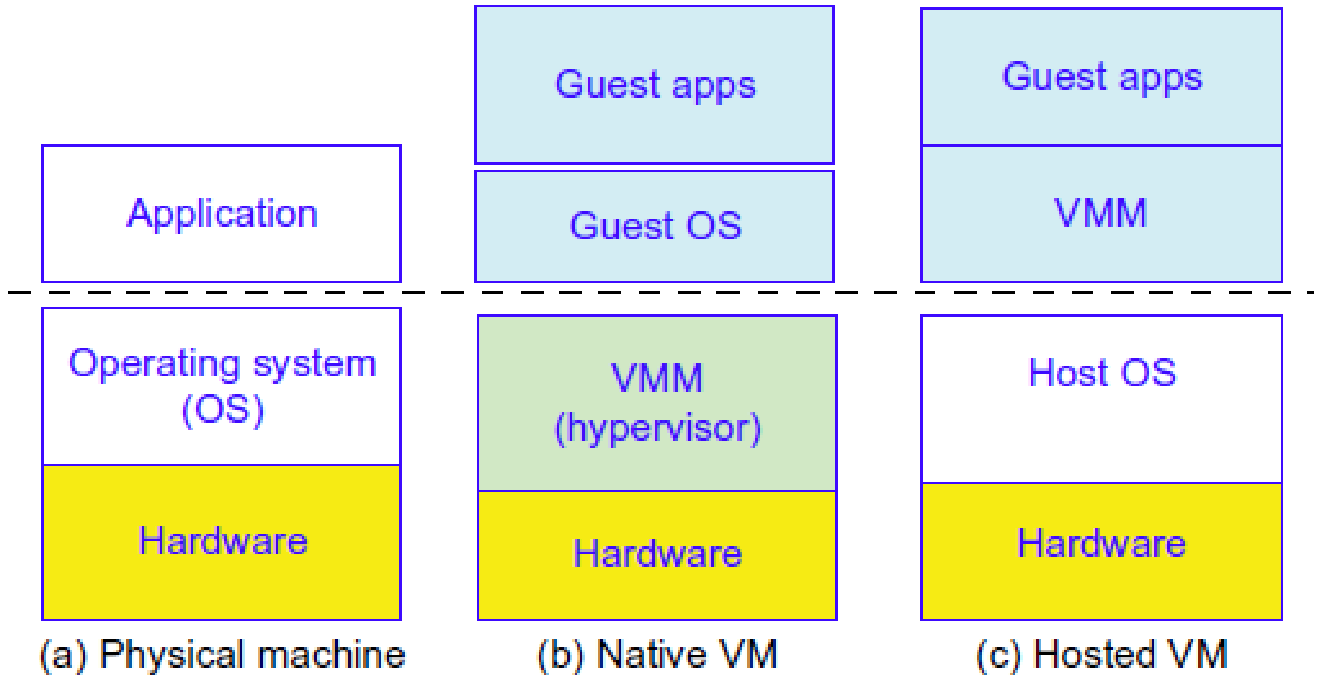
A conventional computer has a single OS image. This offers a rigid architecture that tightly couples application software to a specific hardware platform.

Some software running well on one machine may not be executable on another platform with a different instruction set under a fixed OS.

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

Today, to build large clusters, grids, and clouds, we need to access large amounts of computing, storage, and networking resources in a virtualized manner.

We need to aggregate those resources, and hopefully, offer a single system image. In particular, a cloud of provisioned resources must rely on virtualization of processors, memory, and I/O facilities dynamically .


Application
Operating system (OS)
Hardware
(a) Physical machine
Guest apps
Guest OS
VMM (hypervisor)
Hardware
(b) Native VM
Guest apps
VMM
Host OS
Hardware
(c) Hosted VM

System Models in Network Based Systems

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.

With today's networking technology, a few LAN switches can easily connect hundreds of machines as a working cluster. A WAN can connect many local clusters to form a very large cluster of clusters.

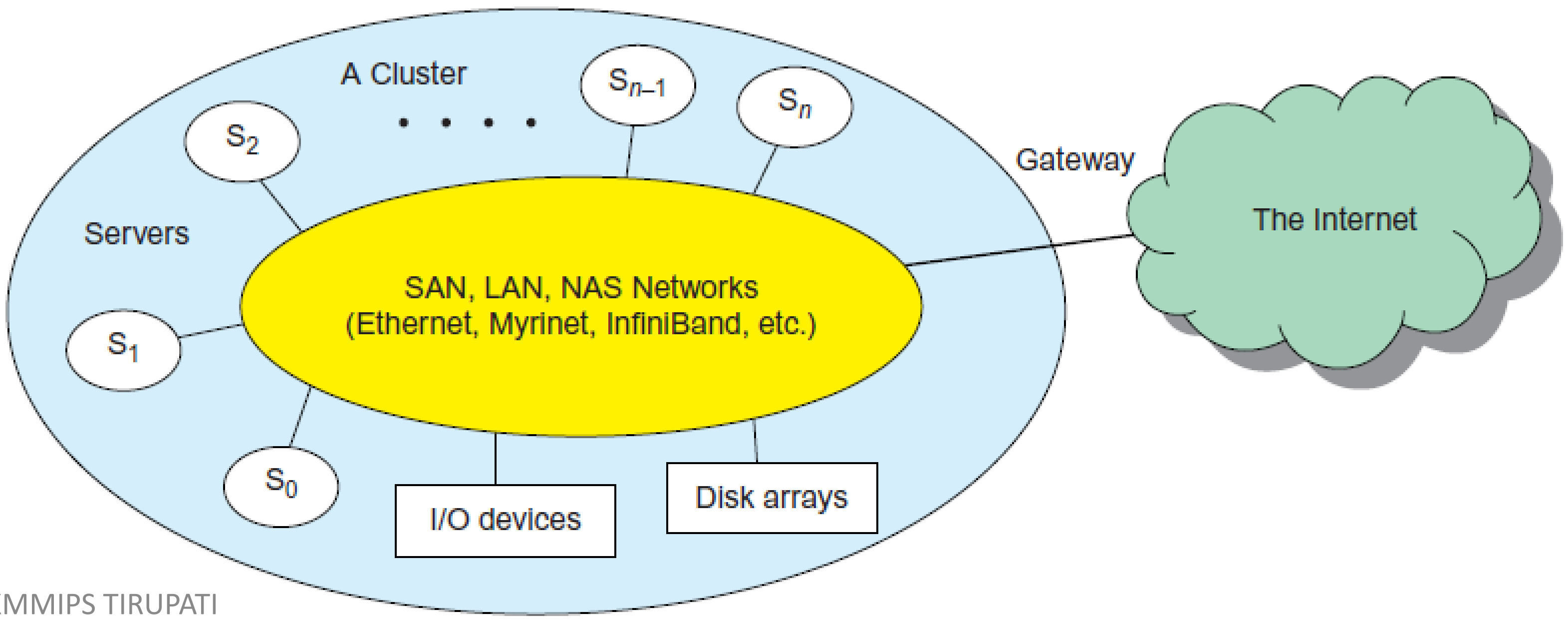
Massive systems are classified into four groups: **clusters, P2P networks, computing grids, and Internet clouds over huge data centers.**

In terms of node number, these four system classes may involve hundreds, thousands, or even millions of computers as participating nodes.

Clusters of Cooperative Computers

- A computing cluster consists of interconnected stand-alone computers which work cooperatively as a single integrated computing resource.
- In the past, clustered computer systems have demonstrated impressive results in handling heavy workloads with large data sets.

Cluster Architecture



- Typical server cluster built around a low-latency, high bandwidth interconnection network.
- This network can be as simple as a SAN (e.g., Myrinet) or aLAN (e.g., Ethernet).

- 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.

- 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.

Single System Image

- An ideal 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.
- SSI makes the cluster appear like a single machine to the user. A cluster with multiple system images is nothing but a collection of independent computers.

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
- Many SSI features are expensive or difficult to achieve at various cluster operational levels.
- Instead of achieving SSI, many clusters are loosely coupled machines.
- Using virtualization, one can build many virtual clusters dynamically, upon user demand.

Major Cluster Design Issues

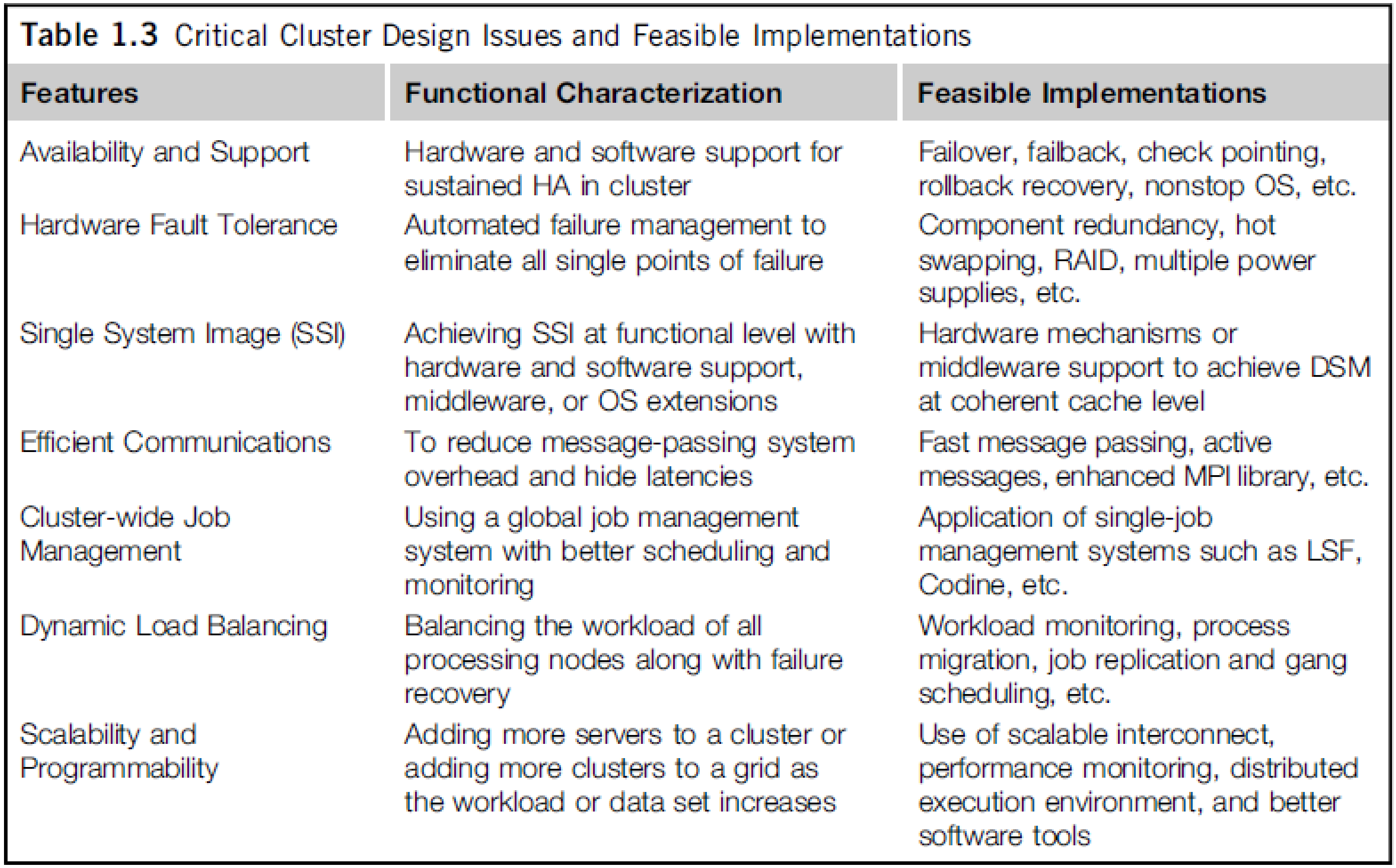


Table 1.3 Critical Cluster Design Issues and Feasible Implementations

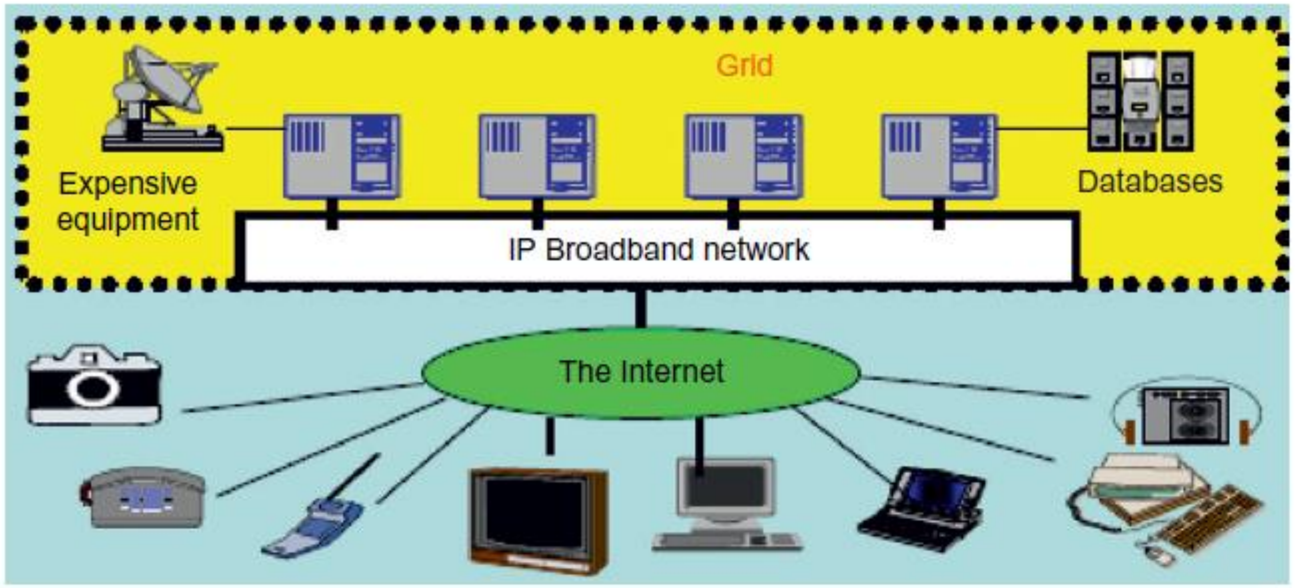
Features	Functional Characterization	Feasible Implementations
Availability and Support	Hardware and software support for sustained HA in cluster	Failover, failback, check pointing, rollback recovery, nonstop OS, etc.
Hardware Fault Tolerance	Automated failure management to eliminate all single points of failure	Component redundancy, hot swapping, RAID, multiple power supplies, etc.
Single System Image (SSI)	Achieving SSI at functional level with hardware and software support, middleware, or OS extensions	Hardware mechanisms or middleware support to achieve DSM at coherent cache level
Efficient Communications	To reduce message-passing system overhead and hide latencies	Fast message passing, active messages, enhanced MPI library, etc.
Cluster-wide Job Management	Using a global job management system with better scheduling and monitoring	Application of single-job management systems such as LSF, Codine, etc.
Dynamic Load Balancing	Balancing the workload of all processing nodes along with failure recovery	Workload monitoring, process migration, job replication and gang scheduling, etc.
Scalability and Programmability	Adding more servers to a cluster or adding more clusters to a grid as the workload or data set increases	Use of scalable interconnect, performance monitoring, distributed execution environment, and better software tools

Grid Computing Infrastructures

- In the past 30 years, users have experienced a natural growth path from Internet to web and grid computing services.
- Internet services such as the Telnet command enables a local computer to connect to a remote computer. A web service such as HTTP enables remote access of remote web pages.
- Grid computing is envisioned to allow close interaction among applications running on distant computers simultaneously.

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.
- Enterprises or organizations present grids as integrated computing resources. The computers used in a grid are primarily workstations, servers, clusters, and supercomputers.
- Personal computers, laptops, and PDAs can be used as access devices to a grid system.



- 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.

- The grid is built across various IP broadband networks including LANs and WANs already used by enterprises or organizations over the Internet.

- The grid is presented to users as an integrated resource pool as shown in the upper half of the figure.

- At the server end, the grid is a network. At the client end, we see wired or wireless terminal devices.

- The grid integrates the computing, communication, contents, and transactions as rented services.

- Enterprises and consumers form the user base, which then defines the usage trends and service characteristics.

- TeraGrid in US, EGEE in Europe, and ChinaGrid in China for various distributed scientific grid applications.

Grid Families

- Grid technology demands new distributed computing models, software/middleware support, network protocols, and hardware infrastructures.

- National grid projects are followed by industrial grid platform development by IBM, Microsoft, Sun, HP, Dell, Cisco, EMC, Platform Computing, and others.

Grid systems are classified in essentially two categories:
computational or data grids and P2P grids.

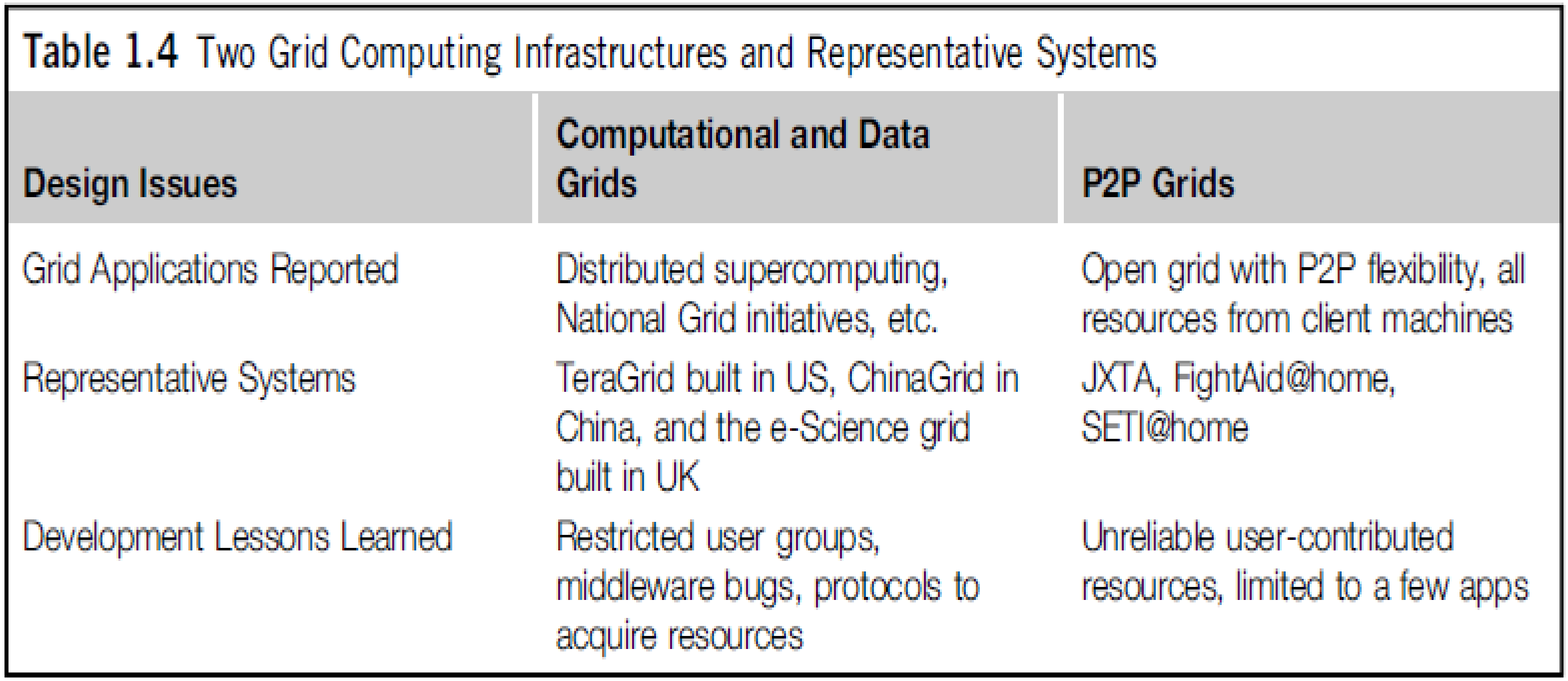


Table 1.4 Two Grid Computing Infrastructures and Representative Systems

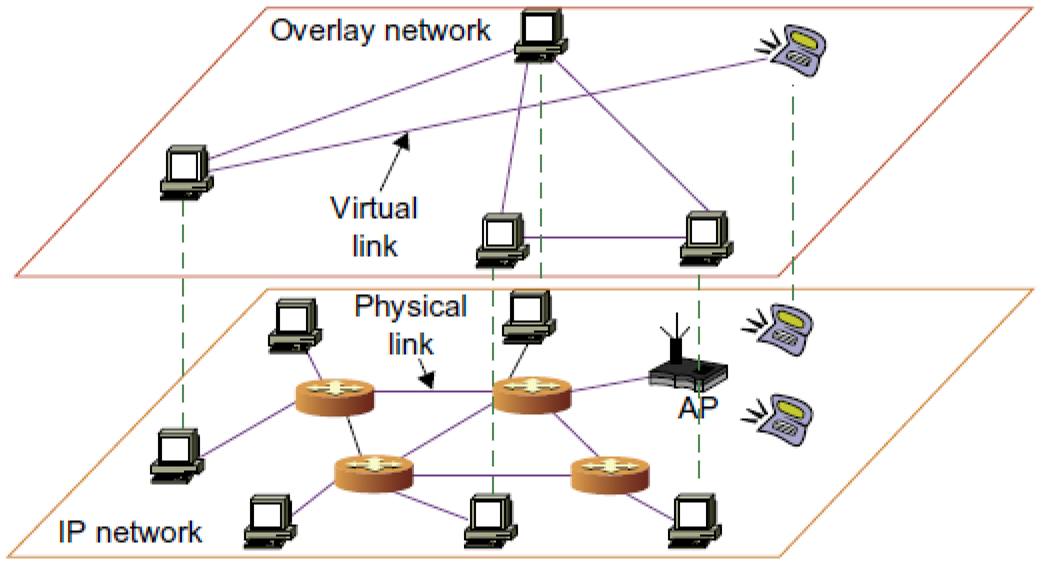
Design Issues	Computational and Data Grids	P2P Grids
Grid Applications Reported	Distributed supercomputing, National Grid initiatives, etc.	Open grid with P2P flexibility, all resources from client machines
Representative Systems	TeraGrid built in US, ChinaGrid in China, and the e-Science grid built 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

Peer-to-Peer Network Families

- An example of a well-established distributed system is the client-server architecture.
- In this scenario, 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.

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.
- In other words, no peer machine has a global view of the entire P2P system. The system is self-organizing with distributed control.



- Architecture of a P2P network at two abstraction levels. Initially, the peers are totally unrelated. Each peer machine joins or leaves the P2P network voluntarily.

- 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.

- Thus, the physical network varies in size and topology dynamically due to the free membership in the P2P network.

- 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 as shown in Figure.

- When a new peer joins the system, its peer ID is added as a node in the overlay network. When an existing peer leaves the system, its peer ID is removed from the overlay network automatically. Therefore, it is the P2P overlay network that characterizes the logical connectivity among the peers.

two types of overlay networks: unstructured and structured.

- An unstructured overlay network is characterized by a random graph. There is no fixed route to send messages or files among the nodes. Often, flooding is applied to send a query to all nodes in an unstructured overlay, thus resulting in heavy network traffic and nondeterministic search results.

- Structured overlay networks follow certain connectivity topology and rules for inserting and removing nodes (peer IDs) from the overlay graph. Routing mechanisms are developed to take advantage of the structured overlays.

P2P Application Families

Based on application, P2P networks are classified into four groups

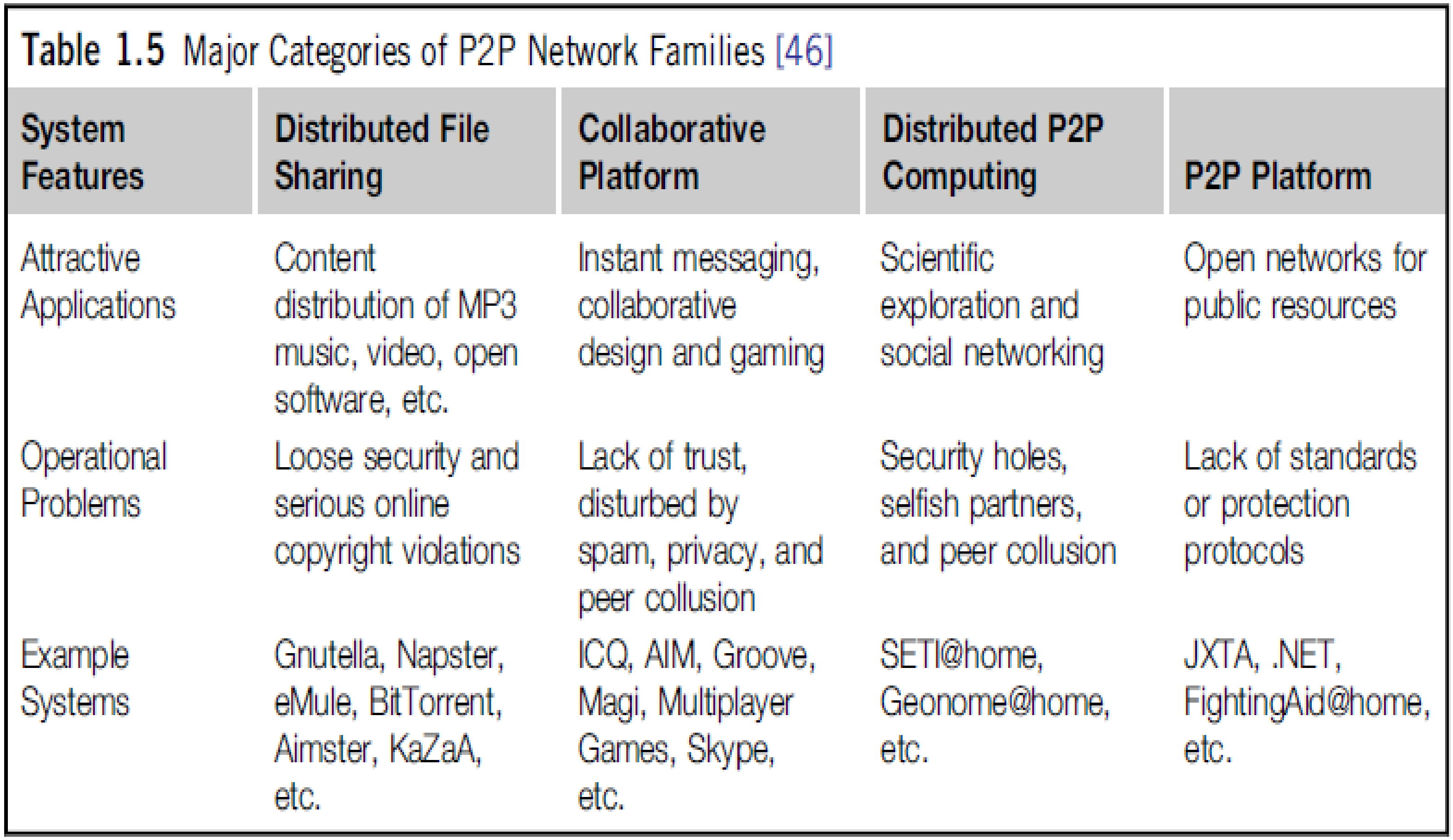


Table 1.5 Major Categories of P2P Network Families [46]

System Features	Distributed File Sharing	Collaborative Platform	Distributed P2P Computing	P2P Platform
Attractive Applications	Content distribution of MP3 music, video, open software, etc.	Instant messaging, collaborative design and gaming	Scientific exploration and social networking	Open networks for public resources
Operational Problems	Loose security and serious online copyright violations	Lack of trust, disturbed by spam, privacy, and peer collusion	Security holes, selfish partners, and peer collusion	Lack of standards or protection protocols
Example Systems	Gnutella, Napster, eMule, BitTorrent, Aimster, KaZaA, etc.	ICQ, AIM, Groove, Magi, Multiplayer Games, Skype, etc.	SETI@home, Geonome@home, etc.	JXTA, .NET, FightingAid@home, etc.

P2P Computing Challenges

- P2P computing faces three types of heterogeneity problems in hardware, software, and network requirements.
- There are too many hardware models and architectures to select from; incompatibility exists between software and the OS; and 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 location is also important to affect collective performance. Data locality, network proximity, and interoperability are three design objectives in distributed P2P applications.
- P2P performance is affected by routing efficiency and self-organization by participating peers.
- Fault tolerance, failure management, and load balancing are other important issues in using overlay networks.
- Security, privacy, and copyright violations are major worries by those in the industry in terms of applying P2P technology in business applications
- the system is not centralized, managing it is difficult

Cloud Computing over the Internet

"Computational science is changing to be data-intensive. Supercomputers must be balanced systems, not just CPU farms but also petascale I/O and networking arrays."

- ➢ 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.

"A cloud is a pool of virtualized computer resources. A cloud can host a variety of different workloads, including batch-style backend jobs and interactive and user-facing applications."

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 .

The idea is to move desktop computing to a service-oriented platform using server clusters and huge databases at data centers.

Cloud computing leverages its low cost and simplicity to benefit both users and providers. Machine virtualization has enabled such cost-effectiveness.

The cloud ecosystem must be designed to be secure, trustworthy, and dependable.

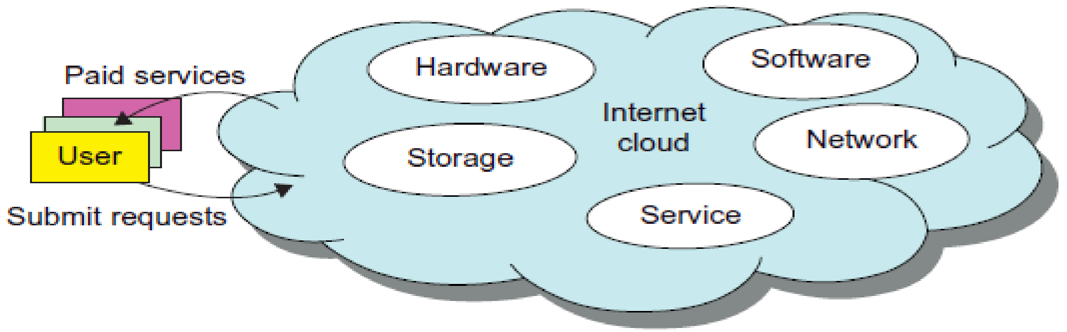


FIGURE 1.18

Virtualized resources from data centers to form an Internet cloud, provisioned with hardware, software, storage, network, and services for paid users to run their applications.

The Cloud Landscape

- Traditionally, a distributed computing system tends to be owned and operated by an autonomous administrative domain (e.g., a research laboratory or company) for on-premises computing needs.
- However, 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. Cloud provides IaaS,PaaS,IaaS services owned by major providers
- Internet clouds offer four deployment modes: private, public, managed, and hybrid

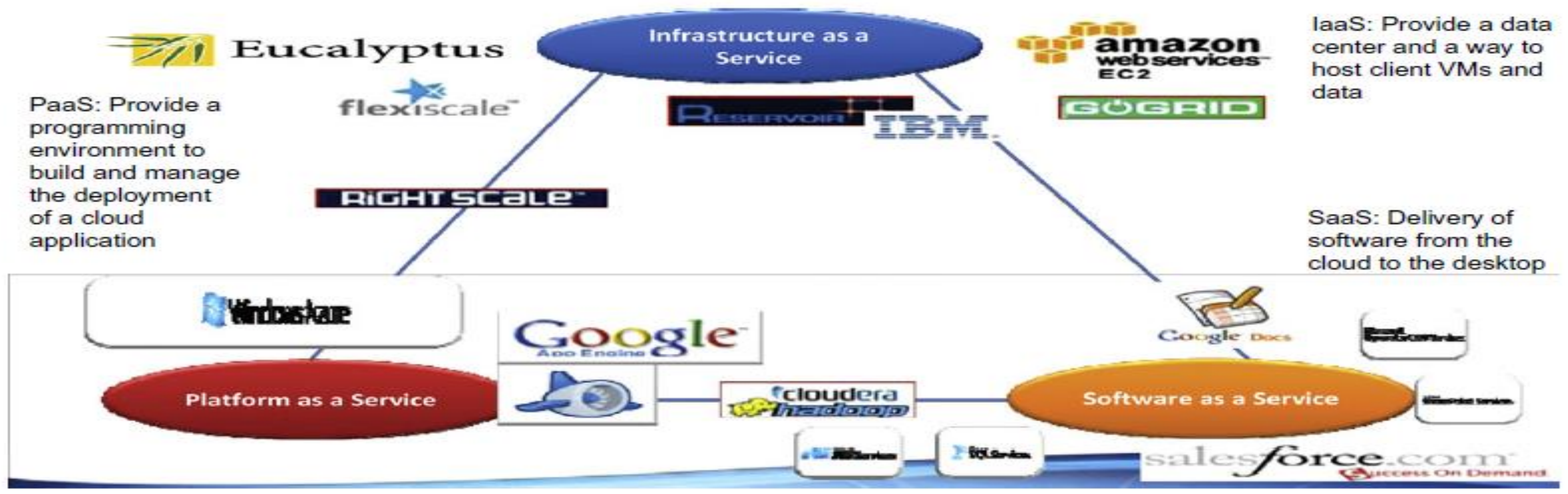

Eucalyptus
Infrastructure as a Service
amazon web services EC2
IaaS: Provide a data center and a way to host client VMs and data
PaaS: Provide a programming environment to build and manage the deployment of a cloud application
flexiscale
Reservoir
IBM
GoGrid
RightScale
SaaS: Delivery of software from the cloud to the desktop
Windows Azure
Google App Engine
Google Docs
Platform as a Service
cloudera hadoop
Software as a Service
salesforce.com Success On Demand

FIGURE 1.19

Three cloud service models in a cloud landscape of major providers.

Following eight reasons to adapt the cloud for upgraded Internet applications and web services:

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

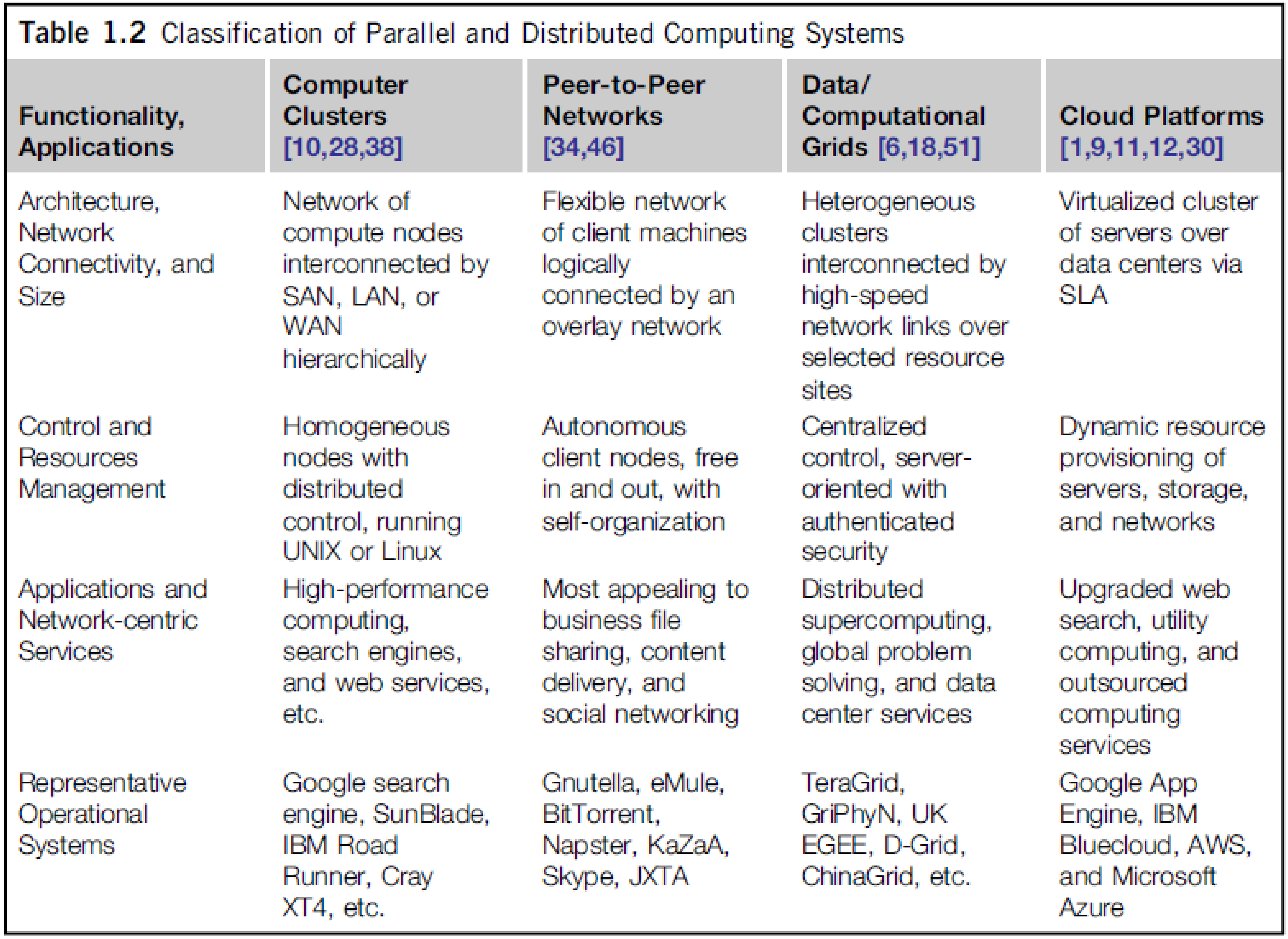


Table 1.2 Classification of Parallel and Distributed Computing Systems

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

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.

Layered Architecture for Web Services and Grids

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 Web-Sphere MQ or Java Message Service (JMS) which provide rich functionality and support virtualization of routing, senders, and recipients.

- **Security** is a critical capability that either uses or reimplements the capabilities seen in concepts such as Internet Protocol Security (IPsec) and secure sockets in the OSI layers.

Discovery and information services

- JNDI (Jini and Java Naming and Directory Interface) illustrating different approaches within the Java distributed object model.
- The CORBA Trading Service, UDDI (Universal Description, Discovery, and Integration),
- LDAP (Lightweight Directory Access Protocol), and ebXML (Electronic Business using eXtensible Markup Language)

Management services include

- service state and lifetime support; examples include the CORBA Life Cycle and Persistent states,
- the different Enterprise JavaBeans models, Jini's lifetime model, and a suite of web services specifications

Web Services and Tools

- Loose coupling and support of heterogeneous implementations make services more attractive than distributed objects.

Two choices of service architecture: web services or REST systems

- In web services, one aims to fully specify all aspects of the service and its environment.
- This specification is carried with communicated messages using Simple Object Access Protocol (SOAP).
- The hosting environment then becomes a universal distributed operating system with fully distributed capability carried by SOAP messages.

In the REST approach, one adopts simplicity as the universal principle and delegates most of the difficult problems to application (implementation-specific) software.

In a web services language, REST has minimal information in the header, and the message body (that is opaque to generic message processing) carries all the needed information. REST architectures are clearly more appropriate for rapid technology environments.

In CORBA and Java, the distributed entities are linked with RPCs, and the simplest way to build composite applications is to view the entities as objects and use the traditional ways of linking them together.

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.

There are three approaches for distributing resource management functions in a distributed computer system.

The **first approach** is to build a network OS over a large number of heterogeneous OS platforms. Such an OS offers the lowest transparency to users, and is essentially a distributed file system, with independent computers relying on file sharing as a means of communication.

The **second approach** is to develop middleware to offer a limited degree of resource sharing , similar to the MOSIX/OS developed for clustered systems

The **third approach** is to develop a truly distributed OS to achieve higher use or system transparency.

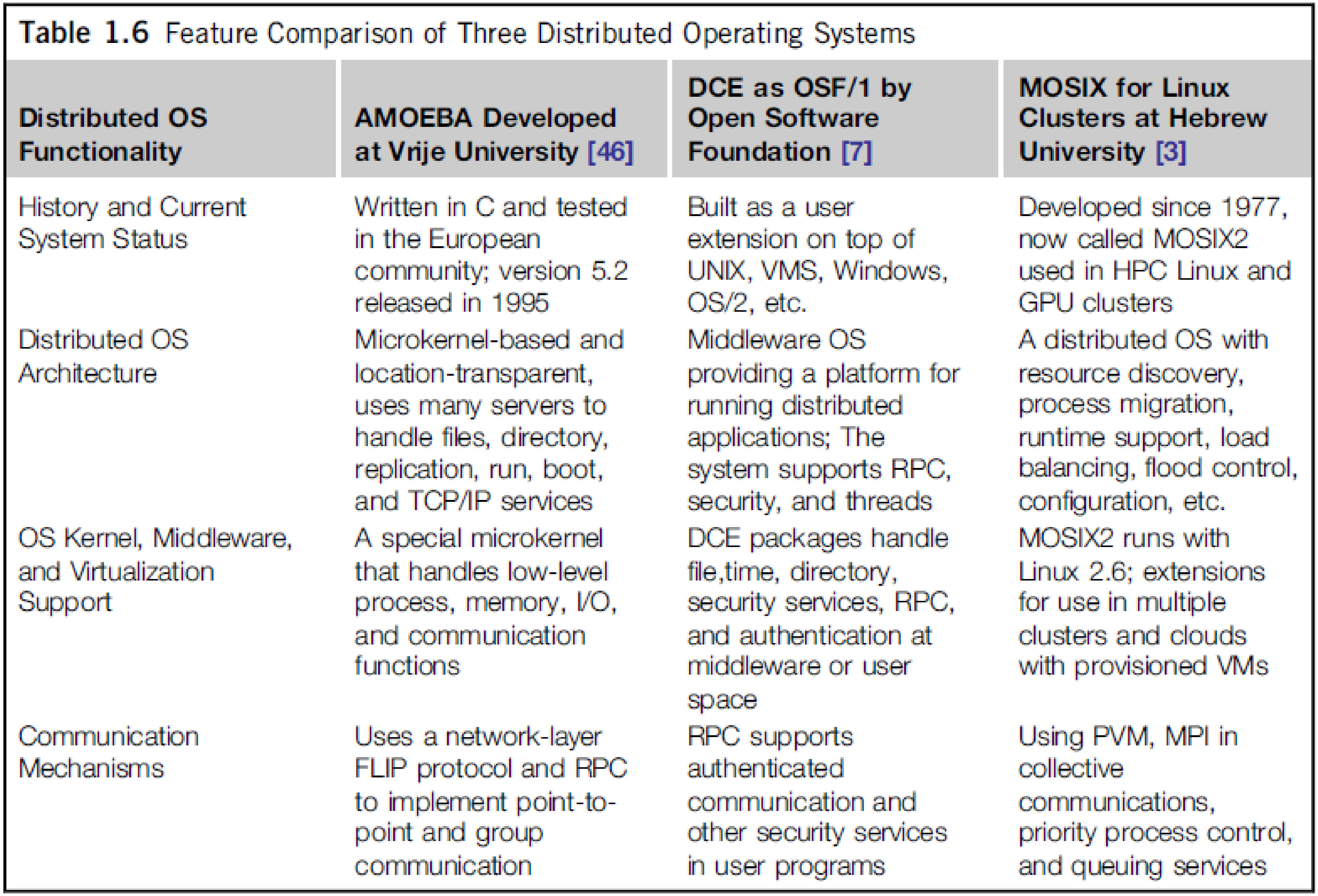


Table 1.6 Feature Comparison of Three Distributed Operating Systems

Distributed OS Functionality	AMOEBA Developed at Vrije University [46]	DCE as OSF/1 by Open Software Foundation [7]	MOSIX for Linux Clusters at Hebrew University [3]
History and Current System Status	Written in C and tested in the European community; version 5.2 released in 1995	Built as a user extension on top of UNIX, VMS, Windows, OS/2, etc.	Developed since 1977, now called MOSIX2 used in HPC Linux and GPU clusters
Distributed OS Architecture	Microkernel-based and location-transparent, uses many servers to handle files, directory, replication, run, boot, and TCP/IP services	Middleware OS providing a platform for running distributed applications; The system supports RPC, security, and threads	A distributed OS with resource discovery, process migration, runtime support, load balancing, flood control, configuration, etc.
OS Kernel, Middleware, and Virtualization Support	A special microkernel that handles low-level process, memory, I/O, and communication functions	DCE packages handle file,time, directory, security services, RPC, and authentication at middleware or user space	MOSIX2 runs with Linux 2.6; extensions for use in multiple clusters and clouds with provisioned VMs
Communication Mechanisms	Uses a network-layer FLIP protocol and RPC to implement point-to-point and group communication	RPC supports authenticated communication and other security services in user programs	Using PVM, MPI in collective communications, priority process control, and queuing services

Parallel and Distributed Programming Models

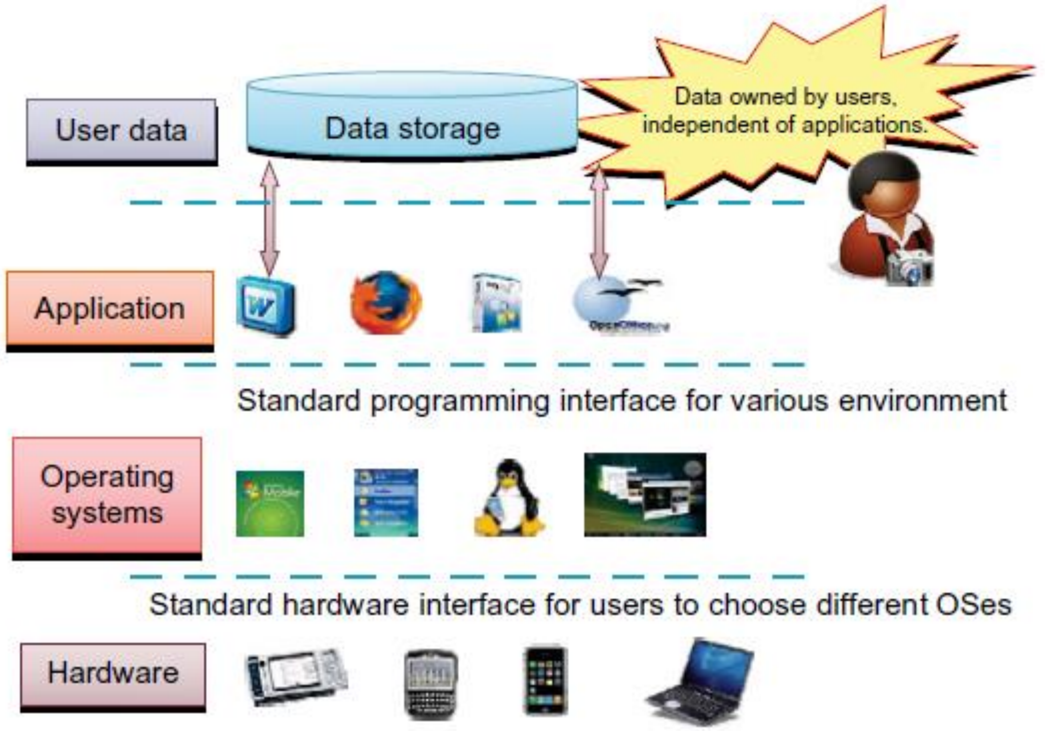

User data
Data storage
Data owned by users, independent of applications.
Application
Standard programming interface for various environment
Operating systems
Standard hardware interface for users to choose different OSes
Hardware

FIGURE 1.22

A transparent computing environment that separates the user data, application, OS, and hardware in time and space – an ideal model for cloud computing.

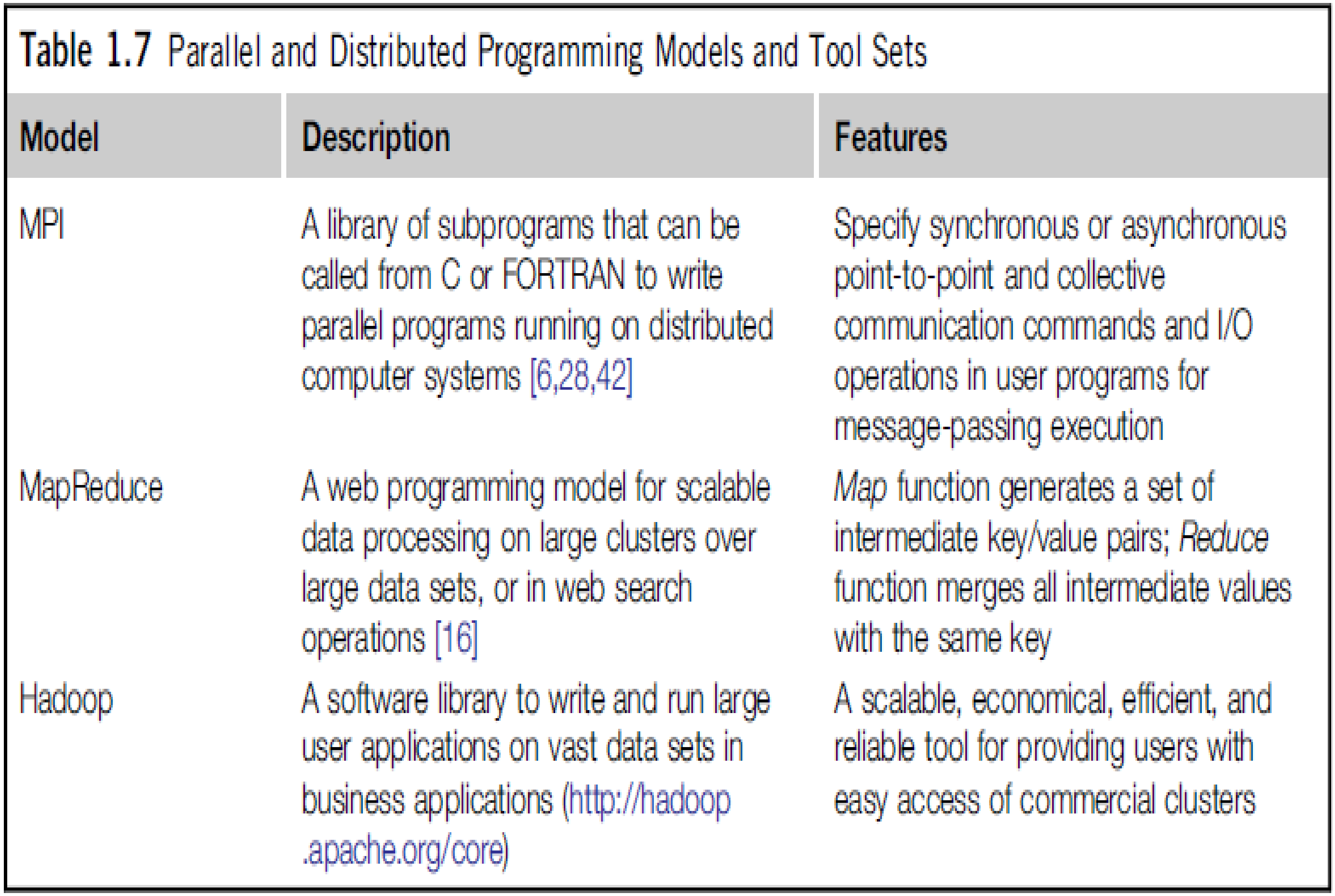


Table 1.7 Parallel and Distributed Programming Models and Tool Sets

Model	Description	Features
MPI	A library of subprograms that can be called from C or FORTRAN to write parallel programs running on distributed computer systems [6,28,42]	Specify synchronous or asynchronous point-to-point and collective communication commands and I/O operations in user programs for message-passing execution
MapReduce	A web programming model for scalable data processing on large clusters over large data sets, or in web search operations [16]	*Map* function generates a set of intermediate key/value pairs; *Reduce* function merges all intermediate values with the same key
Hadoop	A software library to write and run large user applications on vast data sets in business applications (http://hadoop.apache.org/core)	A scalable, economical, efficient, and reliable tool for providing users with easy access of commercial clusters

The idea is to embody clusters, grid systems, and P2P systems with upgraded web services and utility computing applications. Besides **MPI,** distributed programming can be also supported with low-level primitives such as the Parallel Virtual Machine (PVM).

A typical **MapReduce** computation process can handle terabytes of data on tens of thousands or more client machines. Hundreds of MapReduce programs can be executed simultaneously; in fact, thousands of MapReduce jobs are executed on Google's clusters every day.

Hadoop is efficient, as it processes data with a high degree of parallelism across a large number of commodity nodes, and it is reliable in that it automatically keeps multiple data copies to facilitate redeployment of computing tasks upon unexpected system failures.

NIST Cloud Reference Architecture

Cloud definition

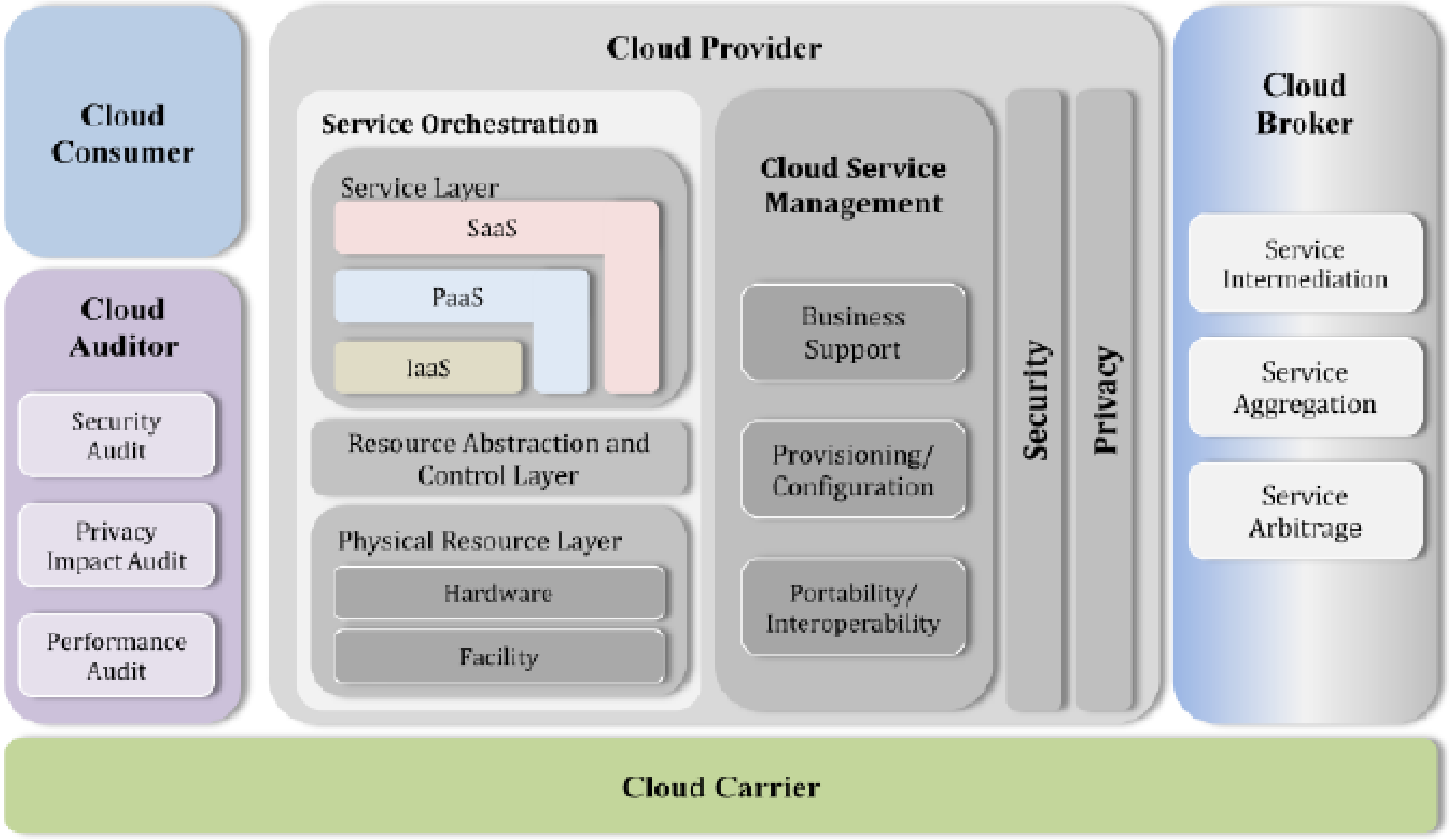
Cloud computing is an information technology service model where computing services (both hardware and software) are delivered on-demand to customers over a network in a self-service fashion, independent of device and location. The resources required to provide the requisite quality-of-service levels are shared, dynamically scalable, rapidly provisioned, virtualized and released with minimal service provider interaction.

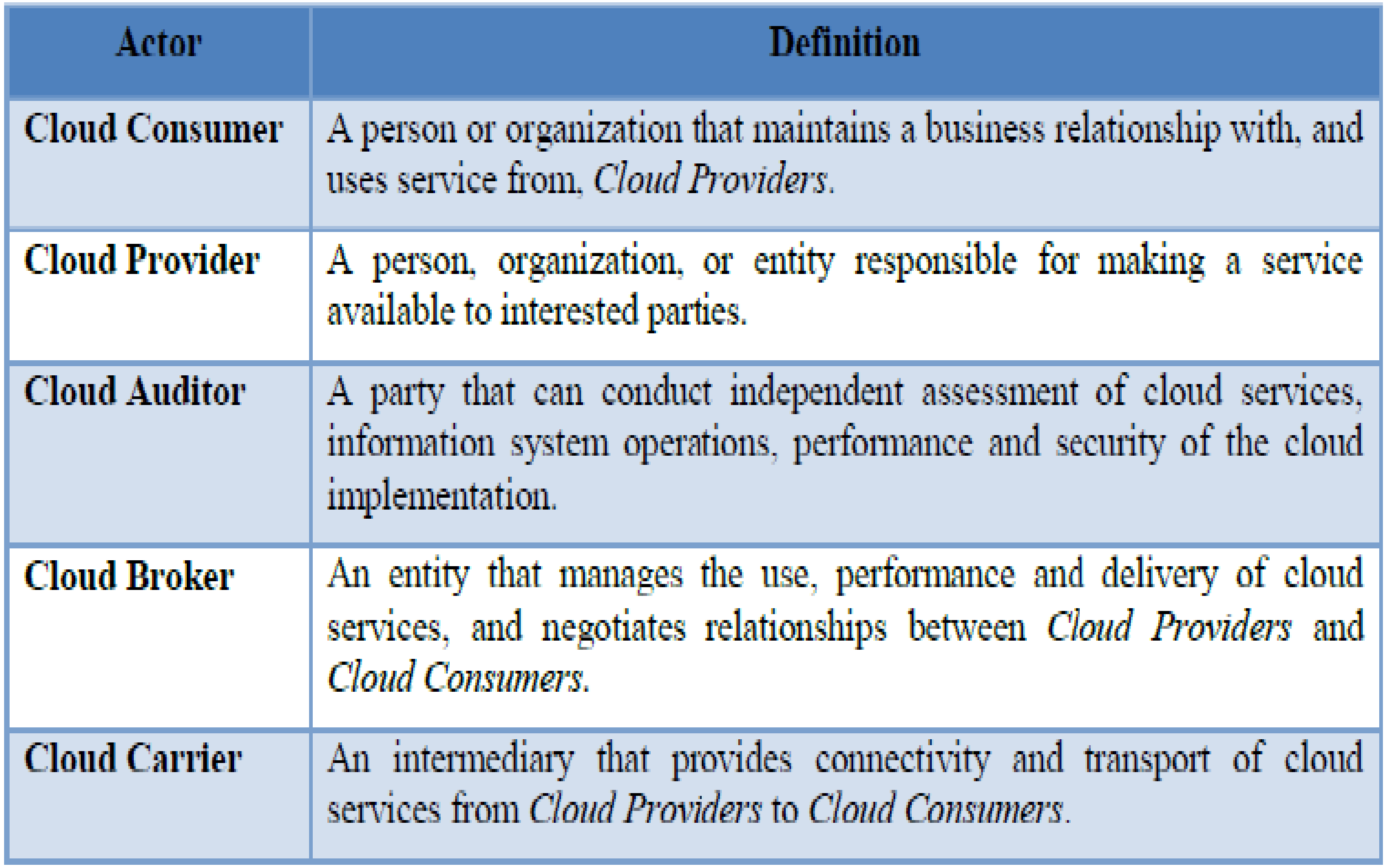
The Figure given below an overview of the NIST cloud computing reference architecture, which identifies the major actors, their activities and functions in cloud computing.

As shown in Figure 1, the NIST cloud computing reference architecture defines five major actors:

- cloud consumer,
- cloud provider,
- cloud carrier,
- cloud auditor
- and cloud broker.

Each actor is an entity (a person or an organization) that participates in a transaction or process and/or performs tasks in cloud computing. Table 1 briefly lists the actors defined in the NIST cloud computing reference architecture.


Cloud Consumer
Cloud Auditor
Security Audit
Privacy Impact Audit
Performance Audit
Cloud Provider
Service Orchestration
Service Layer
SaaS
PaaS
IaaS
Resource Abstraction and Control Layer
Physical Resource Layer
Hardware
Facility
Cloud Service Management
Business Support
Provisioning/ Configuration
Portability/ Interoperability
Security
Privacy
Cloud Broker
Service Intermediation
Service Aggregation
Service Arbitrage
Cloud Carrier



Actor	Definition
Cloud Consumer	A person or organization that maintains a business relationship with, and uses service from, *Cloud Providers*.
Cloud Provider	A person, organization, or entity responsible for making a service available to interested parties.
Cloud Auditor	A party that can conduct independent assessment of cloud services, information system operations, performance and security of the cloud implementation.
Cloud Broker	An entity that manages the use, performance and delivery of cloud services, and negotiates relationships between *Cloud Providers* and *Cloud Consumers*.
Cloud Carrier	An intermediary that provides connectivity and transport of cloud services from *Cloud Providers* to *Cloud Consumers*.

Table 1: Actors in Cloud Computing

Example Usage Scenario 1: A cloud consumer may request service from a cloud broker instead of contacting a cloud provider directly. The cloud broker may create a new service by combining multiple services or by enhancing an existing service. In this example, the actual cloud providers are invisible to the cloud consumer and the cloud consumer interacts directly with the cloud broker.

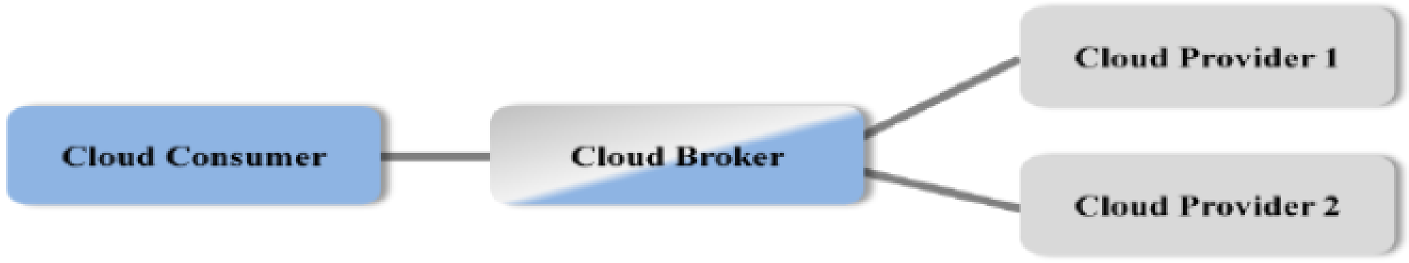
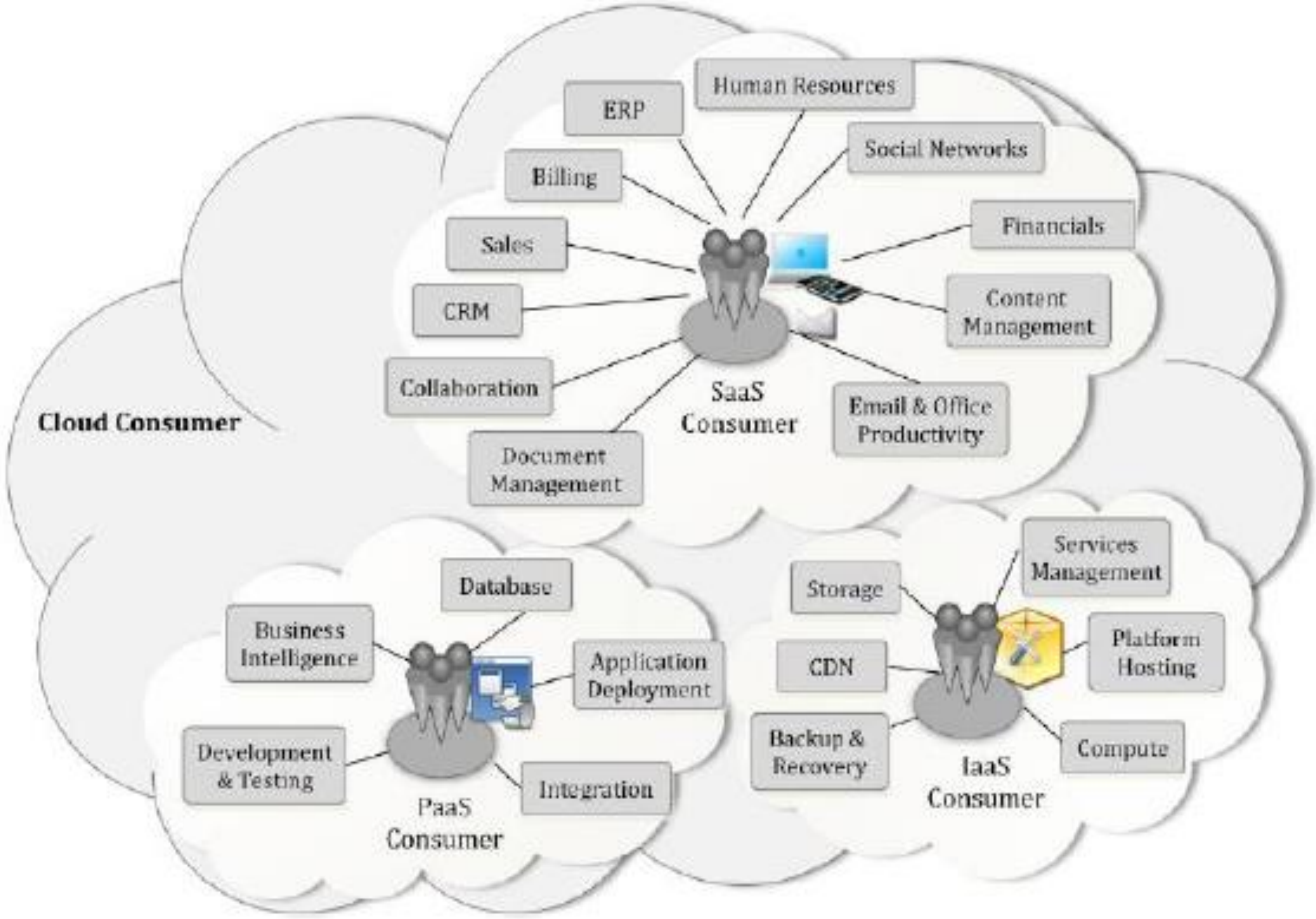


Figure 3: Usage Scenario for Cloud Brokers

Cloud Consumer

The cloud consumer is the principal stakeholder for the cloud computing service. A cloud consumer represents a person or organization that maintains a business relationship with, and uses the service from a cloud provider. A cloud consumer browses the service catalog from a cloud provider, requests the appropriate service, sets up service contracts with the cloud provider, and uses the service.

The cloud consumer may be billed for the service provisioned, and needs to arrange payments accordingly. Cloud consumers need SLAs to specify the technical performance requirements fulfilled by a cloud provider.



Services Available to Cloud Consumer

Cloud provider

cloud provider is a person, an organization; it is the entity responsible for making a service available to interested parties. A Cloud Provider acquires and manages the computing infrastructure required for providing the services, runs the cloud software that provides the services, and makes arrangement to deliver the cloud services to the Cloud Consumers through network access.

For Software as a Service, the cloud provider deploys, configures, maintains and updates the operation of the software applications on a cloud infrastructure so that the services are provisioned at the expected service levels to cloud consumers.

For PaaS, the Cloud Provider manages the computing infrastructure for the platform and runs the cloud software that provides the components of the platform, such as runtime software execution stack, databases, and other middleware components. The PaaS Cloud Provider typically also supports the development, deployment and management process of the PaaS Cloud Consumer by providing tools such as integrated development environments (IDEs), development version of cloud software, software development kits (SDKs), deployment and management tools

.

For IaaS, the Cloud Provider acquires the physical computing resources underlying the service, including the servers, networks, storage and hosting infrastructure. The Cloud Provider runs the cloud software necessary to makes computing resources available to the IaaS Cloud Consumer through a set of service interfaces and computing resource abstractions, such as virtual machines and virtual network interfaces

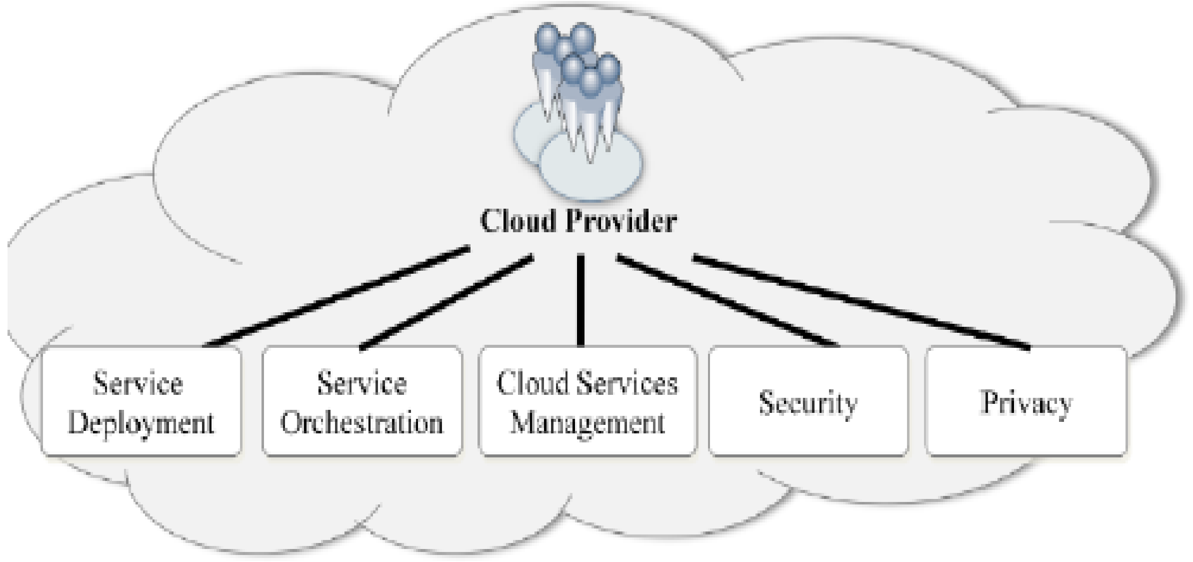


Figure 7: Cloud Provider - Major Activities

Cloud Auditor

A cloud auditor is a party that can perform an independent examination of cloud service controls with the intent to express an opinion thereon. Audits are performed to verify conformance to standards through review of objective evidence. A cloud auditor can evaluate the services provided by a cloud provider in terms of security controls, privacy impact, performance, etc.

Cloud Broker

As cloud computing evolves, the integration of cloud services can be too complex for cloud consumers to manage. A cloud consumer may request cloud services from a cloud broker, instead of contacting a cloud provider directly. A cloud broker is an entity that manages the use, performance and delivery of cloud services and negotiates relationships between cloud providers and cloud consumers.

Service Intermediation: A cloud broker enhances a given service by improving some specific capability and providing value-added services to cloud consumers.
Service Aggregation: A cloud broker combines and integrates multiple services into one or more new services.
Service Arbitrage: Service arbitrage is similar to service aggregation except that the services being aggregated are not fixed

Cloud Carrier

A cloud carrier acts as an intermediary that provides connectivity and transport of cloud services between cloud consumers and cloud providers.

Cloud carriers provide access to consumers through network, telecommunication and other access devices. For example, cloud consumers can obtain cloud services through network access devices, such as computers, laptops, mobile phones, mobile Internet devices (MIDs).

Service management, also known as information technology **service management**, **software** allows companies to manage how it provides **services** to its customers. **Services** include order **management**, hardware and **software** maintenance, diagnostics and troubleshooting, as well as routine operations.

Key operations include :

Customer accounts.
Provisioning and configuring resources.
Service catalogs.
Performance.
Security.
Interoperability and portability.

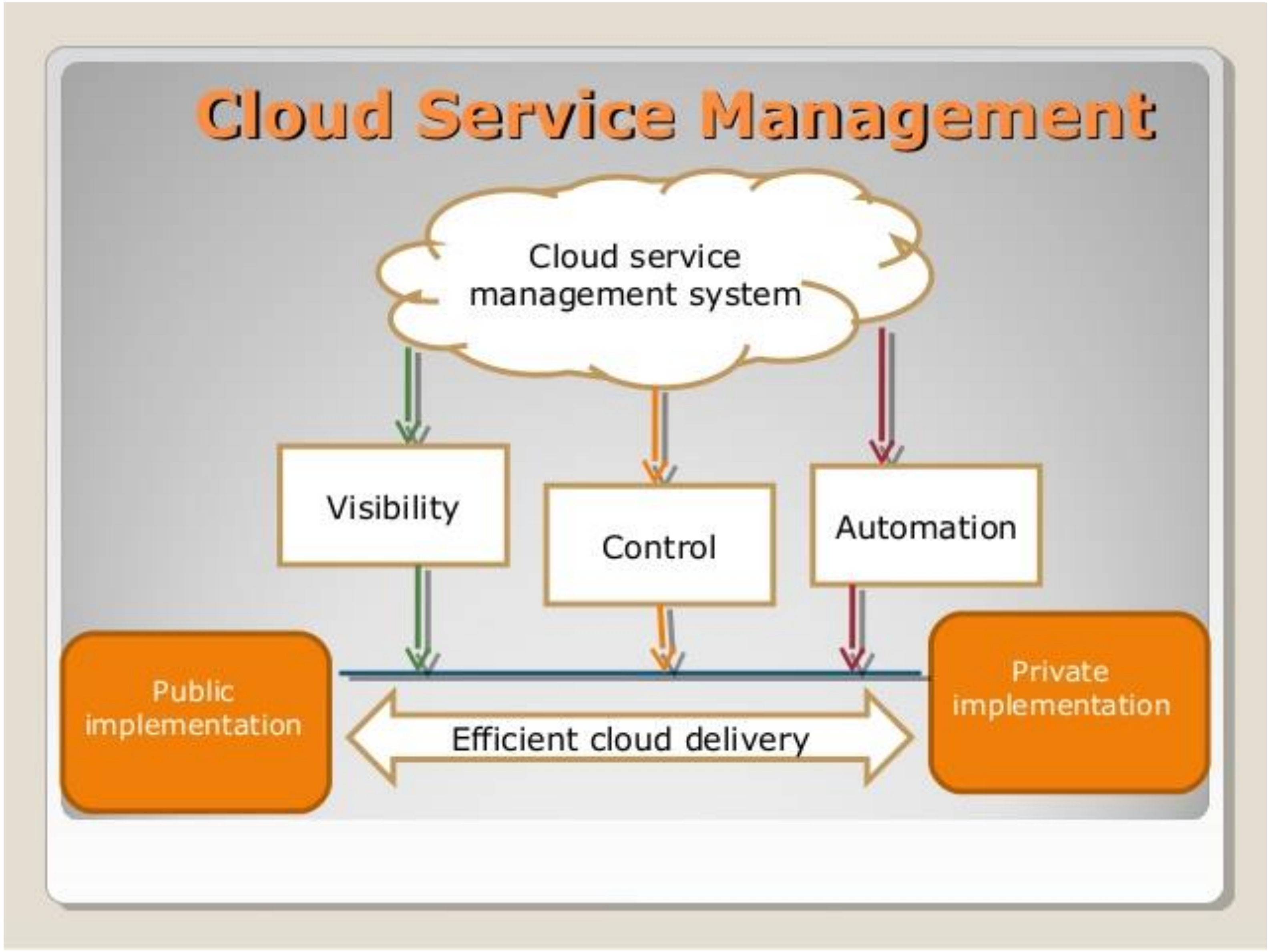
Service Management in Cloud

- The level of responsibility for managing the delivery and operation of cloud services in cloud computing depends on your role in the cloud ecosystem.
- Because the hybrid cloud encompasses so many different services from so many different places, cloud management needs lots of attention.
- Secure, reliable, interoperable, and cost-effective cloud services require an underlying infrastructure that is highly tuned, well monitored, and well managed.

Write about cloud Provider, cloud consumer, from above question.,SaaS,PaaS,IaaS

An organization consuming Software as a Service (SaaS), Platform as a Service (PaaS), Business Process as a Service (BPaaS), and Infrastructure as a Service (IaaS) from different vendors may need to play multiple roles, including both cloud consumer and cloud provider.

Responsibilities for managing and securing these cloud services will overlap and often be shared across different participants.


Cloud Service Management
Cloud service management system
Visibility
Control
Automation
Public implementation
Efficient cloud delivery
Private implementation

Cloud providers have responsibility for the architecture of the cloud and for providing cloud resources and services to cloud consumers. These organizations are responsible for monitoring the cloud infrastructure to ensure that service levels for uptime, speed of delivery, and other performance characteristics are maintained.

In order to bring the right level of management to cloud environments, a cloud provider needs to understand and anticipate the requirements of its customers, provide the right level of security and governance, optimize operating costs, and ensure that cloud services are available to users. Cloud providers also assume the responsibility of quickly finding the root cause of any problems that interfere with the quality of service delivery to maintain compliance with service-level agreements.

Cloud management responsibilities may increase exponentially as use of cloud services expands. As both the quantity and type of cloud services used across your organization increase, it becomes challenging to maintain a consistent level of visibility and control.

A cloud service broker is a vendor or a business unit within a company that provides oversight and management for a variety of cloud providers. The broker can stand between an organization and its cloud services, managing the use, performance, integration, and delivery of such resources.