ARUNAI ENGINEERING COLLEGE

DEPARTMENT OF CSE

CS8791 CLOUD COMPUTING

IV YEAR – VII SEMESTER

LECTURE NOTES

UNIT I INTRODUCTION

Introduction to Cloud Computing – Definition of Cloud – Evolution of Cloud Computing – Underlying Principles of Parallel and Distributed Computing – Cloud Characteristics – Elasticity in Cloud – On-demand Provisioning.

INTRODUCTION

EVOLUTION OF DISTRIBUTED COMPUTING

Grids enable access to shared computing power and storage capacity from your desktop. Clouds enable access to leased computing power and storage capacity from your desktop.

- Grids are an **open source** technology. Resource users and providers alike can understand and contribute to the management of their grid
- Clouds are a **proprietary** technology. Only the resource provider knows exactly how their cloud manages data, job queues, security requirements and so on.
- The concept of grids was proposed in 1995. The Open science grid (OSG) started in 1995 The EDG (European Data Grid) project began in 2001.
- In the late 1990`s Oracle and EMC offered early private cloud solutions . However the term cloud computing didn't gain prominence until 2007.

SCALABLE COMPUTING OVER THE INTERNET

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

- 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.

The Platform Evolution

- From 1950 to 1970, a handful of mainframes, including the IBM 360 and CDC 6400

- From 1960 to 1980, lower-cost minicomputers such as the DEC PDP 11 and VAX Series
- 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

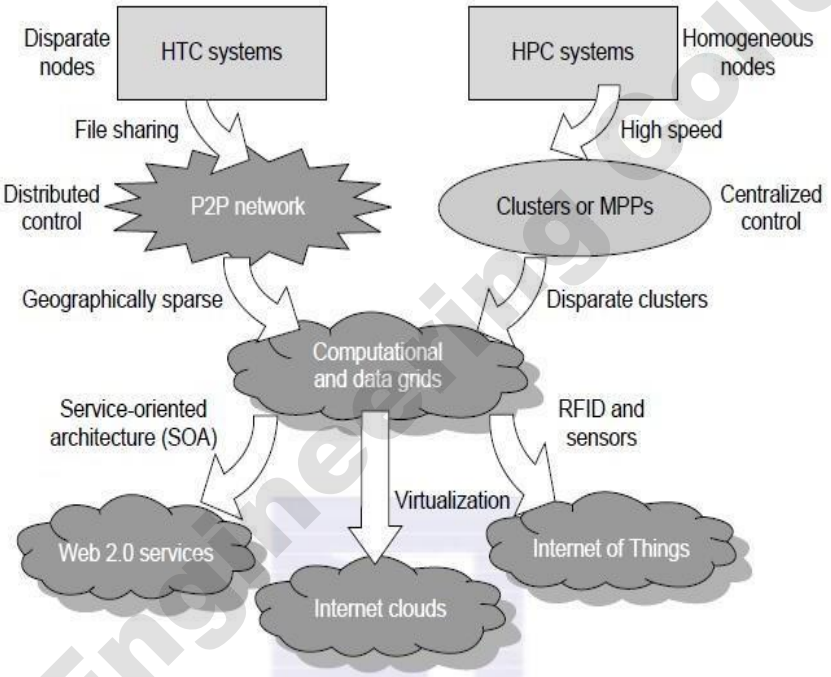


FIGURE 1.1

Evolutionary trend toward parallel, distributed, and cloud computing with clusters, MPPs, P2P networks, grids, clouds, web services, and the Internet of Things.

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.

- 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 (a concept we will discuss further in Chapter 5). Peer machines are globally distributed in nature. P2P, cloud computing, and web service platforms are more focused on

HTC applications than on HPC applications. Clustering and P2P technologies lead to the development of computational grids or data grids.

- 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.
- 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.
- Advances in virtualization make it possible to see the growth of Internet clouds as a new computing paradigm. The maturity of radio-frequency identification (RFID), Global Positioning System (GPS), and sensor technologies has triggered the development of the Internet of Things (IoT). These new paradigms are only briefly introduced here.
- 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, and cloud computing overlaps with distributed, centralized, and parallel computing.

Terms

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 parallel computing, all processors are either tightly coupled with centralized shared memory or loosely coupled with distributed memory. Inter processor communication is accomplished through shared memory or via message passing. Acomputer system capable of parallel computing is commonly known as a parallel computer. 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** 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. 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 . As an alternative to the preceding terms, some in the high-tech community prefer the term concurrent computing or concurrent programming. These terms typically refer to the union of parallel computing and distributing computing, although biased practitioners may interpret them differently.
- **Ubiquitous computing** refers to computing with pervasive devices at any place and time using wired or wireless communication. The Internet of Things (IoT) is a networked connection of everyday objects including computers, sensors, humans, etc. The IoT is supported by Internet clouds to achieve ubiquitous computing with any object at any place and time. Finally, the term Internet computing is even broader and covers all computing paradigms over the Internet. This book covers all the aforementioned computing paradigms, placing more emphasis on distributed and cloud computing and their working systems, including the clusters, grids, P2P, and cloud systems.

Internet of Things

- The traditional Internet connects machines to machines or web pages to web pages. The concept of the IoT was introduced in 1999 at MIT .

- 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.
- It allows objects to be sensed and controlled remotely across existing network infrastructure

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. 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.

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

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

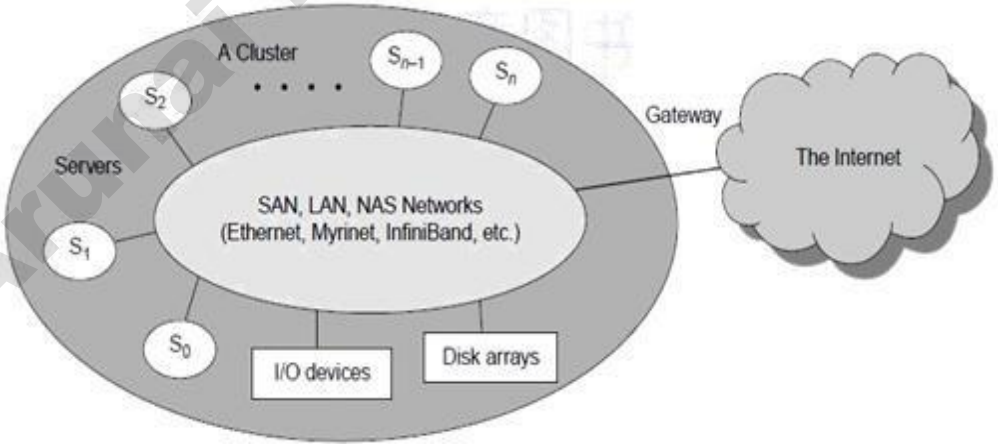


Figure 1.2 Clusters of Servers

Figure 1.2shows the architecture of a 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 a LAN (e.g., Ethernet).

- To build a larger cluster with more nodes, the interconnection network can be built with multiple levels of Gigabit Ethernet, 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.

Single-System Image(SSI)

- 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, 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. For example, distributed memory has multiple images. Users may want all distributed memory to be shared by all servers by forming distributed shared

memory (DSM). 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.

Cloud Computing over the Internet

- 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.
- 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.

a. 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. Cloud computing intends to satisfy many user applications simultaneously.

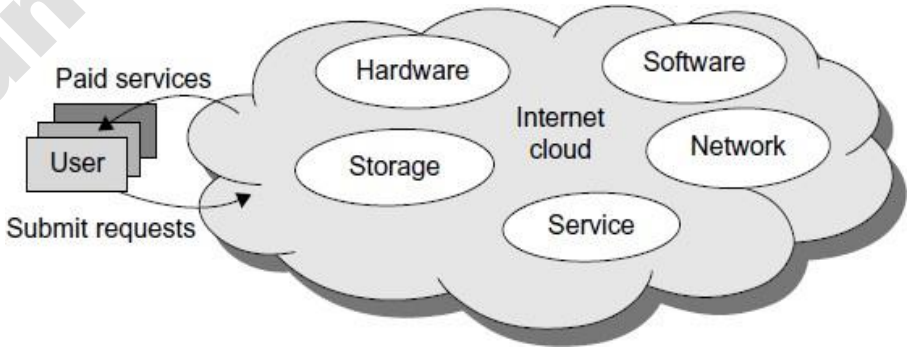


Figure 1.3 Internet Cloud

b. The Cloud Landscape

- The cloud ecosystem must be designed to be secure, trustworthy, and dependable. Some computer users think of the cloud as a centralized resource pool. Others consider the cloud to be a server cluster which practices distributed computing over all the servers 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.
- Cloud computing as an on-demand computing paradigm resolves or relieves us from these problems.

Three Cloud service Model in a cloud landscape

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 OS 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.

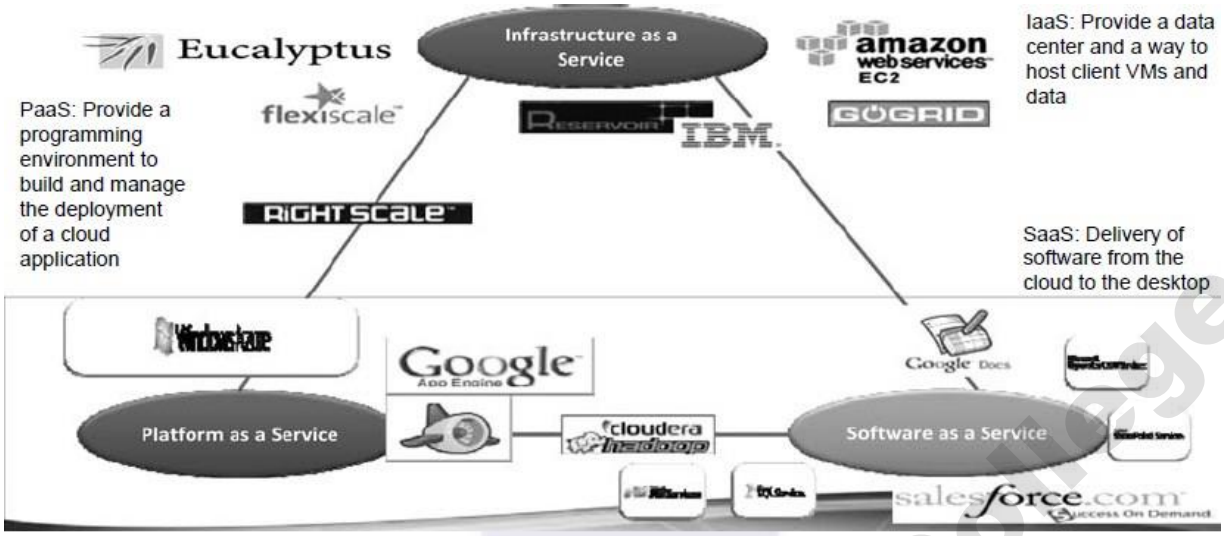


Figure 1.4 The Cloud Landscape in an application

Internet clouds offer four deployment modes: private, public, managed, and hybrid . These modes demand different levels of security implications. The different SLAs imply that the security responsibility is shared among all the cloud providers, the cloud resource consumers, and the third party cloud-enabled software providers. Advantages of cloud computing have been advocated by many IT experts, industry leaders, and computer science researchers.

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

- Cloud computing is using the internet to access someone else's software running on someone else's hardware in someone else's data center.
- The user sees only one resource (HW, Os) but uses virtually multiple os. HW resources etc..
- Cloud architecture effectively uses virtualization
- A model of computation and data storage based on “pay as you go” access to “unlimited” remote data center capabilities
- A cloud infrastructure provides a framework to manage scalable, reliable, on-demand access to applications
- Cloud services provide the “invisible” backend to many of our mobile applications
- High level of elasticity in consumption
- Historical roots in today’s Internet apps
    - Search, email, social networks, e-com sites
    - File storage (Live Mesh, Mobile Me)

Definition

- **“The National Institute of Standards and Technology (NIST) defines cloud computing as a "pay-per-use model for enabling available, convenient and on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction."**

Cloud Computing Architecture

- Architecture consists of 3 tiers
    - Cloud Deployment Model
    - Cloud Service Model
    - Essential Characteristics of Cloud Computing

Essential Characteristics 1

- On-demand self-service.

- A consumer can unilaterally provision computing capabilities such as server time and network storage as needed automatically, without requiring human interaction with a service provider.

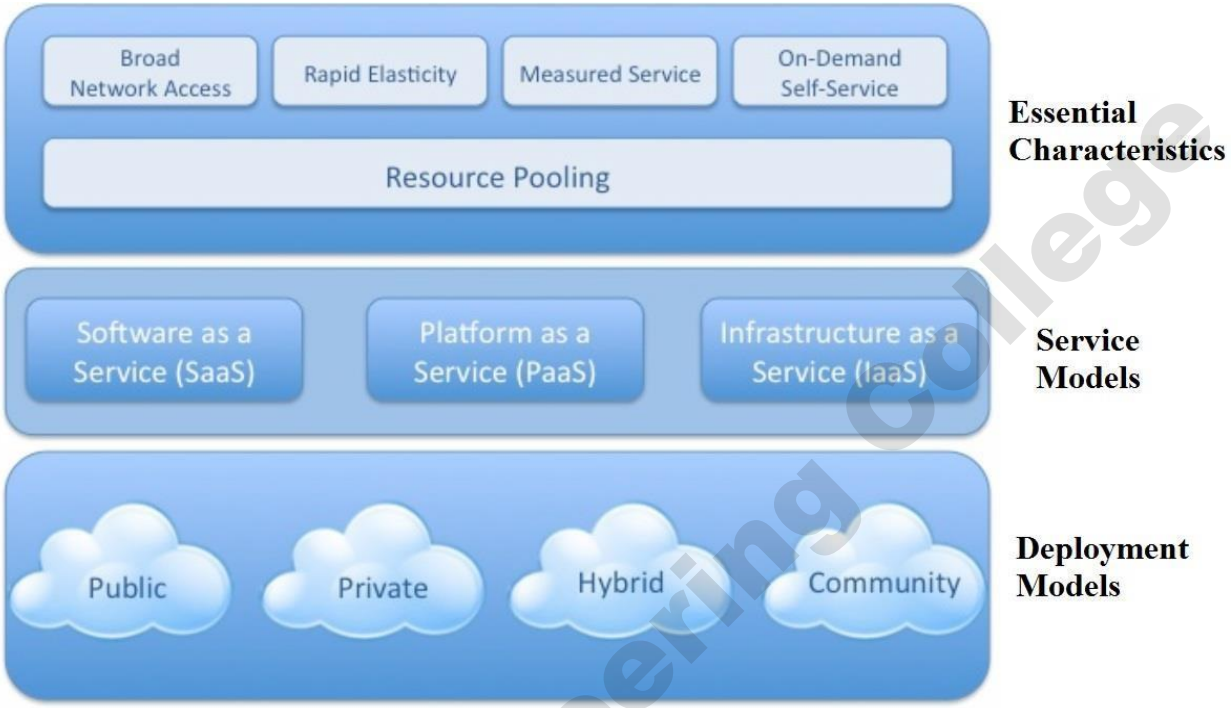


Figure 1.5 Cloud Computing Architecture

Essential Characteristics 2

- Broad network access.
  - Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, laptops, and PDAs) as well as other traditional or cloudbased software services.

Essential Characteristics 3

- Resource pooling.
  - The provider's computing resources are pooled to serve multiple consumers using a **multi-tenant model**, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand.

Essential Characteristics 4

- **Rapid elasticity.**
  - Capabilities can be rapidly and elastically provisioned - in some cases automatically - to quickly scale out; and rapidly released to quickly scale in.
  - To the consumer, the capabilities available for provisioning often appear to be unlimited and can be purchased in any quantity at any time.

Essential Characteristics 5

- **Measured service.**
  - Cloud systems automatically control and optimize resource usage by leveraging a metering capability at some level of abstraction appropriate to the type of service.
  - Resource usage can be monitored, controlled, and reported - providing transparency for both the provider and consumer of the service.

Cloud Service Models

- Cloud **S**oftware as a Service (**S**aaS)
- Cloud **P**latform as a Service (**P**aaS)
- Cloud **I**nfrastructure as a Service (**I**aaS)

SaaS

- SaaS is a licensed software offering on the cloud and pay per use
- SaaS is a software delivery methodology that provides licensed multi-tenant access to software and its functions remotely as a Web-based service. Usually billed based on usage
  - Usually multi tenant environment
  - Highly scalable architecture
- Customers do not invest on software application programs
- The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure.
- The applications are accessible from various client devices through a thin client interface such as a web browser (e.g., web-based email).

- The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, data or even individual application capabilities, with the possible exception of limited user specific application configuration settings.

SaaS providers

- Google's Gmail, Docs, Talk etc
- Microsoft's Hotmail, Sharepoint
- SalesForce,
- Yahoo, Facebook

Infrastructure as a Service (IaaS)

- IaaS is the delivery of technology infrastructure (mostly hardware) as an on demand, scalable service
  - Usually billed based on usage
  - Usually multi tenant virtualized environment
  - Can be coupled with Managed Services for **OS** and application support
  - User can choose his OS, storage, deployed app, networking components
  -

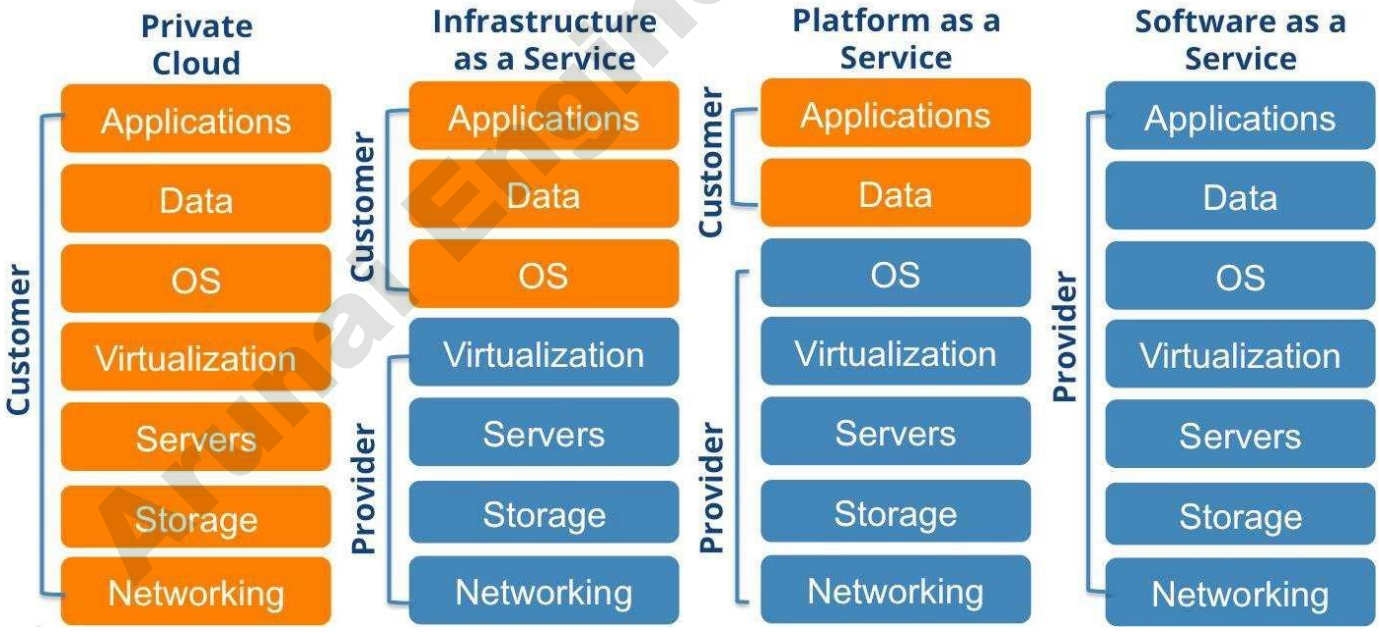


Figure 1.6 Cloud Service Model

- The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources.

- Consumer is able to deploy and run arbitrary software, which may include operating systems and applications.
- The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly limited control of select networking components (e.g., host firewalls).

IaaS providers

- Amazon Elastic Compute Cloud (EC2)
  - Each instance provides 1-20 processors, upto 16 GB RAM, 1.69TB storage
- RackSpace Hosting
  - Each instance provides 4 core CPU, upto 8 GB RAM, 480 GB storage
- Joyent Cloud
  - Each instance provides 8 CPUs, upto 32 GB RAM, 48 GB storage
- Go Grid
  - Each instance provides 1-6 processors, upto 15 GB RAM, 1.69TB storage

Platform as a Service (PaaS)

- PaaS provides all of the facilities required to support the complete life cycle of building, delivering and deploying web applications and services entirely from the Internet. Typically applications must be developed with a particular platform in mind
  - Multi tenant environments
  - Highly scalable multi tier architecture
- The capability provided to the consumer is to deploy onto the cloud infrastructure consumer created or acquired applications created using programming languages and tools supported by the provider.
- The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly application hosting environment configurations.

PaaS providers

- Google App Engine
  - Python, Java, Eclipse

- Microsoft Azure
  - .Net, Visual Studio
- Sales Force
  - Apex, Web wizard
- TIBCO,
- VMware,
- Zoho

Cloud Computing - Opportunities and Challenges

- It enables services to be used without any understanding of their infrastructure.
- Cloud computing works using economies of scale
- It potentially lowers the outlay expense for start up companies, as they would no longer need to buy their own software or servers.
- Cost would be by on-demand pricing.
- Vendors and Service providers claim costs by establishing an ongoing revenue stream.
- Data and services are stored remotely but accessible from "anywhere"

Cloud Computing – Pros

- Lower computer costs
- Instant software updates:
  - When the application is web-based, updates happen automatically
- Improved document format compatibility
- e capacity:
  - Cloud computing offers virtually limitless storage
  - • Increased data reliability:

Cloud Computing – Cons

- Need of Internet :
  - A dead Internet connection means no work and in areas where Internet connections are few or inherently unreliable, this could be a deal-breaker.
  - Requires a constant Internet connection

- Can be slow:
  - Even with a fast connection, web-based applications can sometimes be slower than accessing a similar software program on your desktop PC.
- Disparate Protocols :
  - Each cloud systems uses different protocols and different APIs – Standards yet to evolve.

Evolution of Cloud Computing

Evolution of Cloud Computing

- Cloud Computing Leverages dynamic resources to deliver a large number of services to end users.
- It is High Throughput Computing(HTC) paradigm
- It enables users to share access to resources from anywhere at any time

II Hardware Evolution

- In 1930, binary arithmetic was developed
  - computer processing technology, terminology, and programming languages.

• In 1939,Electronic computer was developed

- Computations were performed using vacuum-tube technology.

• In 1941, Konrad Zuse's Z3 was developed

- Support both floating-point and binary arithmetic.

There are four generations

- First Generation Computers
- Second Generation Computers
- Third Generation Computers
- Fourth Generation Computers

a.First Generation Computers

Time Period : 1942 to 1955

Technology : Vacuum Tubes

Size : Very Large System

Processing : Very Slow

Examples:

1.ENIAC (Electronic Numerical Integrator and Computer)

2.EDVAC(Electronic Discrete Variable Automatic Computer)

Advantages:

• It made use of vacuum tubes which was the advanced technology at that time

• Computations were performed in milliseconds.

Disadvantages:

• very big in size, weight was about 30 tones.

• very costly.

• Requires more power consumption

•Large amount heat was generated.

b.Second Generation Computers

Time Period : 1956 to 1965.

Technology : Transistors

Size : Smaller

Processing : Faster

o Examples

Honeywell 400

IBM 7094

Advantages

- Less heat than first generation.
- Assembly language and punch cards were used for input.
- Low cost than first generation computers.
- Computations was performed in microseconds.
- Better Portability as compared to first generation

Disadvantages:

- A cooling system was required.
- Constant maintenance was required.
- Only used for specific purposes

c.Third Generation Computers

Time Period : 1966 to 1975

Technology : ICs (Integrated Circuits)

Size : Small as compared to 2nd generation computers

Processing : Faster than 2nd generation computers

Examples

• PDP-8 (Programmed Data Processor)

• PDP-11

Advantages

• These computers were cheaper as compared to generation computers.

• They were fast and reliable.

• IC not only reduce the size of the computer but it also improves the performance of the computer

• Computations was performed in nanoseconds

Disadvantages

• IC chips are difficult to maintain.

• The highly sophisticated technology required for the manufacturing of IC chips.

•Air Conditioning is required

d.Fourth Generation Computers

Time Period : 1975 to Till Date

Technology : Microprocessor

Size : Small as compared to third generation computer

Processing : Faster than third generation computer

Examples

• IBM 4341

• DEC 10

Advantages:

- Fastest in computation and size get reduced as compared to the previous generation of computer. Heat generated is small.
- Less maintenance is required.

Disadvantages:

- The Microprocessor design and fabrication are very complex.
- Air Conditioning is required in many cases

III Internet Hardware Evolution

- Internet Protocol is the standard communications protocol used by every computer on the Internet.
- The conceptual foundation for creation of the Internet was significantly developed by three individuals.
  - Vannevar Bush — MEMIX (1930)
  - Norbert Wiener
  - Marshall McLuhan
- Licklider was founder for the creation of the AR PANET (Advanced Research Projects Agency Network)
- Clark deployed a minicomputer called an Interface Message Processor (IMP) at each site.
- Network Control Program (NCP)- first networking protocol that was used on the ARPANET

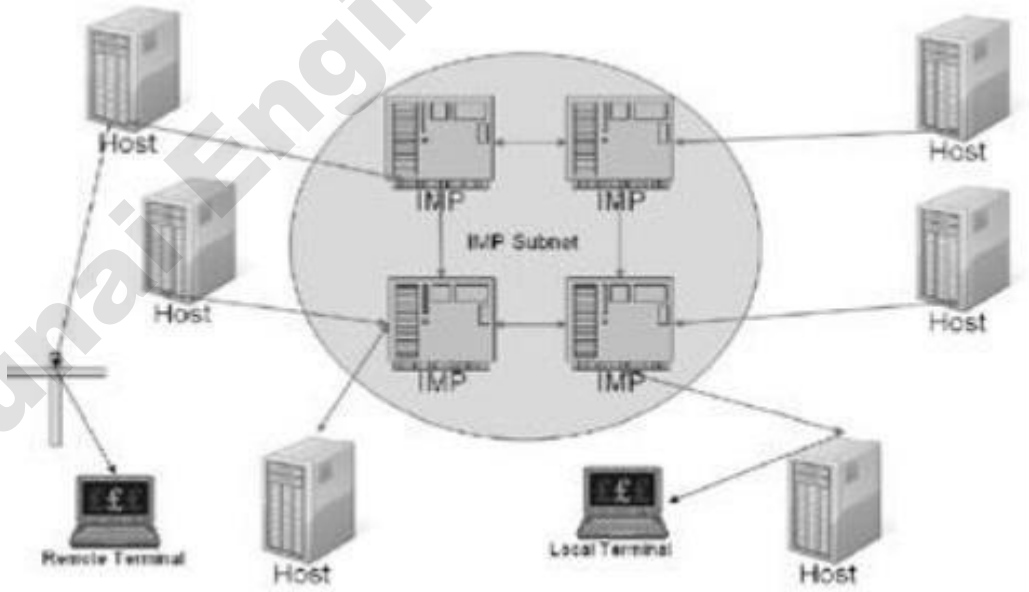


Figure 1.7 **IMP Architecture**

Internet Hardware Evolution

- Establishing a Common Protocol for the Internet
- Evolution of Ipv6
- Finding a Common Method to Communicate Using the Internet Protocol
- Building a Common Interface to the Internet
- The Appearance of Cloud Formations From One Computer to a Grid of Many

a.Establishing a Common Protocol for the Internet

- NCP essentially provided a transport layer consisting of the ARPANET Host-to-Host Protocol (AIIIIP) and the Initial Connection Protocol (ICP)
- Application protocols
  - File Transfer Protocol (FTP), used for file transfers,
  - Simple Mail Transfer Protocol (SMTP), used for sending email

Four versions of TCP/IP

- TCP vl
- TCP v2
- TCP v3 and IP v3,
- TCP v4 and IP v4

b.Evolution of Ipv6

- IPv4 was never designed to scale to global levels.
- To increase available address space, it had to process large data packets (i.e., more bits of data).
- To overcome these problems, Internet Engineering Task Force (IETF) developed IPv6, which was released in January 1995.
- Ipv6 is sometimes called the Next Generation Internet Protocol (IPNG) or TCP/IP v6.

c.Finding a Common Method to Communicate Using the Internet Protocol

- In the 1960s,the word ktpertext was created by Ted Nelson.
- In 1962, Engelbart's first project was Augment, and its purpose was to develop computer tools to augment human capabilities.
- He developed the mouse, Graphical user interface (GUI), and the first working hypertext system, named NLS (oN-Line System).

- NLS was designed to cross-reference research papers for sharing among geographically distributed researchers.
- In the 1980s, Web was developed in Europe by Tim Berners-Lee and Robert Cailliau

d.Building a Common Interface to the Internet

- Betters-Lee developed the first web browser featuring an integrated editor that could create hypertext documents.
- Following this initial success, Berners-Lee enhanced the server and browser by adding support for the FTP (File Transfer protocol)

Figure 1.8 First Web Browser

- Mosaic was the first widely popular web browser available to the general public. Mosaic support for graphics, sound, and video clips.
- In October 1994, Netscape released the first beta version of its browser, Mozilla 0.96b, over the Internet.
- In 1995, Microsoft Internet Explorer was developed that supports both a graphical Web browser and the name for a set of technologies.
- Mozilla Firefox. released in November 2004, became very popular almost immediately.

e.The Appearance of Cloud Formations From One Computer to a Grid of Many

- Two decades ago, computers were clustered together to form a single larger computer in order to simulate a supercomputer and greater processing power.
- In the early 1990s, Ian Foster and Carl Kesselman presented their concept of "The Grid." They used an analogy to the electricity grid, where users could plug in and use a (metered) utility service.
- A major problem in clustering model was data residency. Because of the distributed nature of a grid, computational nodes could be anywhere in the world.

- The Globus Toolkit is an open source software toolkit used for building grid systems and applications

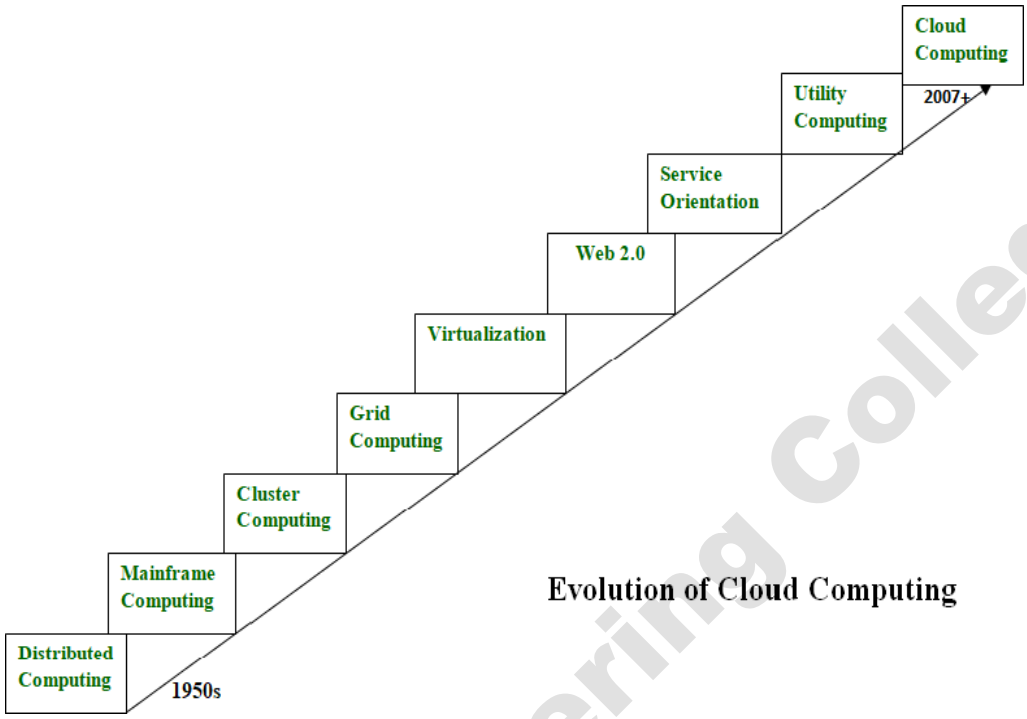


Figure 1.9 Evolution

Evolution of Cloud Services

2008-2009	**Google Application Engine** **Microsoft Azure**
2006	**S3 launches EC2**
2002	**Launch of Amazon Web Services**
1990	**The first milestone of cloud computing arrival of salesforce.com**
1960	**Super Computers** **Mainframes**

IV. SERVER VIRTUALIZATION

- Virtualization is a method of running multiple independent virtual operating systems on a single physical computer.
- This approach maximizes the return on investment for the computer.

- Virtualization technology is a way of reducing the majority of hardware acquisition and maintenance costs, which can result in significant savings for any company.
    - Parallel Processing
    - Vector Processing
    - Symmetric Multiprocessing Systems
    - Massively Parallel Processing Systems

a.Parallel Processing

- Parallel processing is performed by the simultaneous execution of program instructions that have been allocated across multiple processors.
- Objective: running a progran in less time.
- The next advancement in parallel processing-multiprogramming
- In a multiprogramming system, multiple programs submitted by users but each allowed to use the processor for a short time.
- This approach is known as "round-robin scheduling"(RR scheduling)

b.Vector Processing

- Vector processing was developed to increase processing performance by operating in a multitasking manner.
- Matrix operations were added to computers to perform arithmetic operations.
- This was valuable in certain types of applications in which data occurred in the form of vectors or matrices.
- In applications with less well-formed data, vector processing was less valuable.

c.Symmetric Multiprocessing Systems

- Symmetric multiprocessing systems (SMP) was developed to address the problem of resource management in master/slave models.
- In SMP systems, each processor is equally capable and responsible for managing the workflow as it passes through the system.
- The primary goal is to achieve sequential consistency

d.Massively Parallel Processing Systems

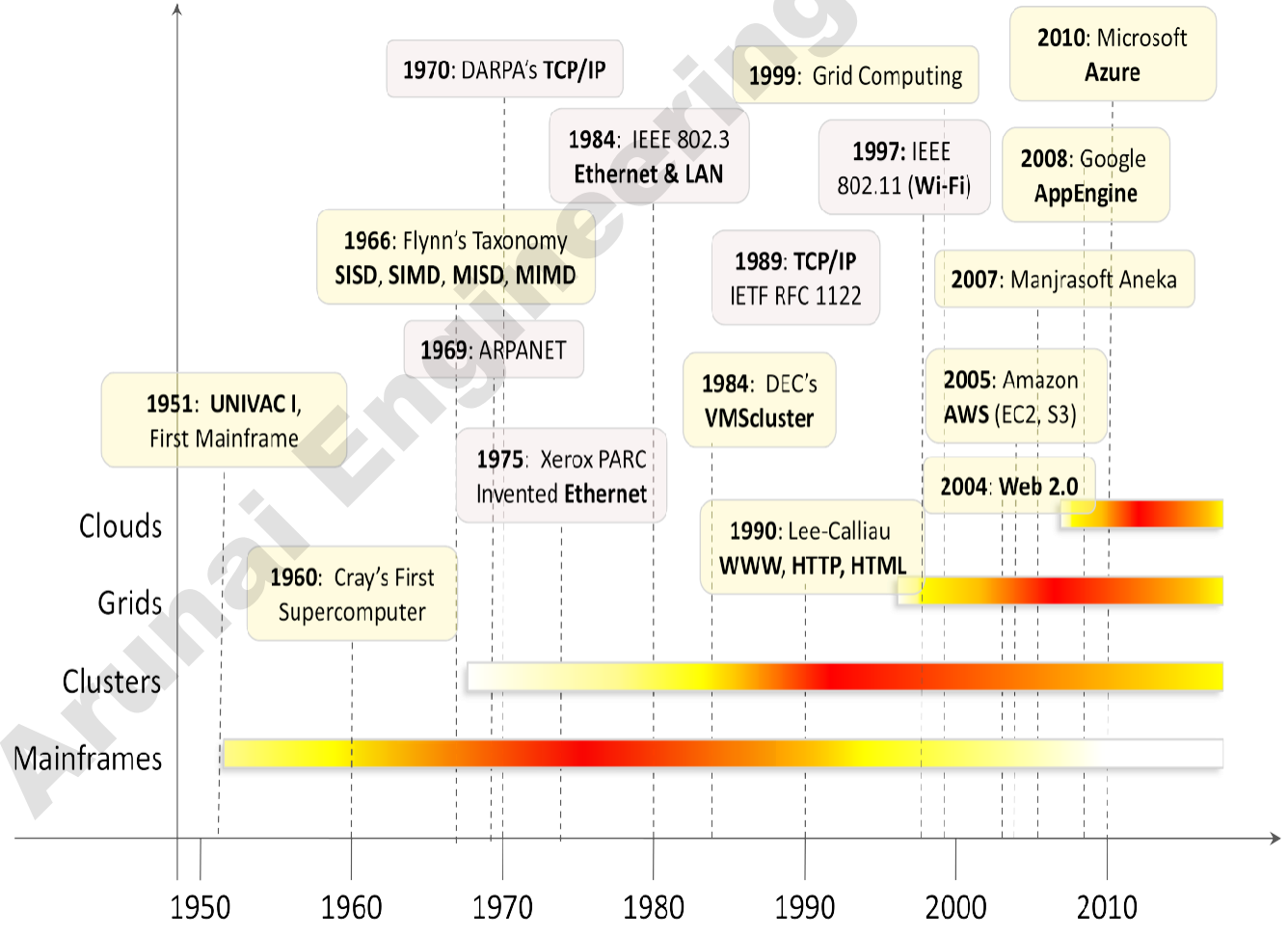
- In Massively Parallel Processing Systems, a computer system with many independent arithmetic units, which run in parallel.
- All the processing elements are interconnected to act as one very large computer.

- Early examples of MPP systems were the Distributed ArrayProcessor, the Goodyear MPP, the Connection Machine, and the Ultracomputer
- MPP machines are not easy to program, but for certain applications, such as data mining, they are the best solution

Principles of Parallel and Distributed Computing

- **Three major milestones have led to cloud computing evolution**
  - Mainframes: Large computational facilities leveraging multiple processing units. Even though mainframes cannot be considered as distributed systems, they offered large computational power by using multiple processors, which were presented as a single entity to users.

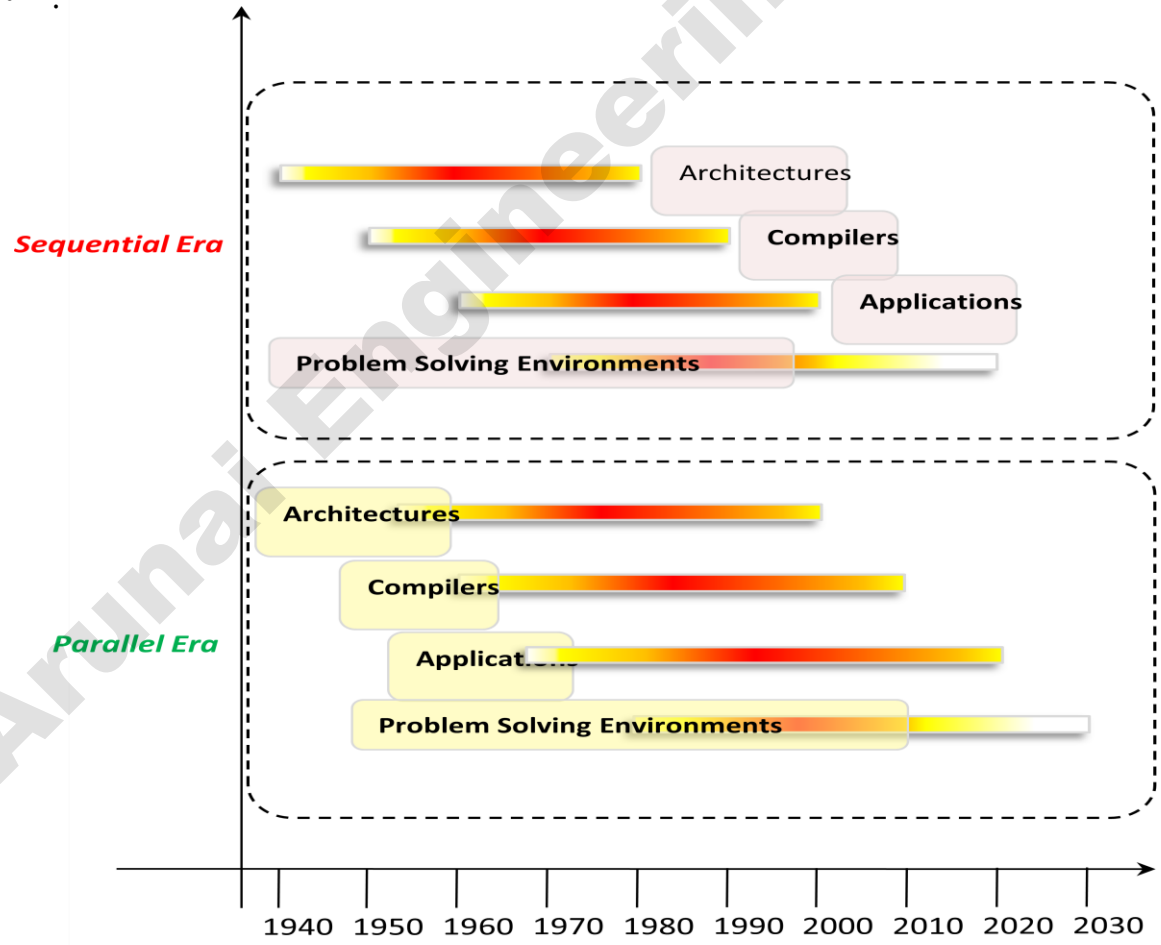
Mile Stones to Cloud computing Evolution



  - Clusters: An alternative technological advancement to the use of mainframes and super computers.
  - Grids
- Clouds

Eras of Computing

- Two fundamental and dominant models of computing are ***sequential*** and ***parallel***.
  - The sequential era began in the 1940s, and Parallel(and distributed) computing era followed it within a decade.
- Four key elements of computing developed during three eras are
  - Architecture
  - Compilers
  - Applications
  - Problem solving environments
- .



- The computing era started with development in ***hardware architectures***, which actually enabled the creation of ***system software*** – particularly in the area of **compilers and operating systems** – which support the management of such systems and the development of ***applications***

- The term parallel computing and distributed computing are **often used interchangeably**, even though they mean **slightly different things**.
- The term **parallel implies a tightly coupled system**, where as **distributed systems refers to a wider class of system, including those that are tightly coupled**.
- More precisely, the term **parallel computing** refers to a model in which **the computation is divided among several processors sharing the same memory**.
- The architecture of **parallel computing system** is often characterized by the **homogeneity of components**: **each processor is of the same type and it has the same capability as the others.**

- The shared memory has a single address space, which is accessible to all the processors.
- Parallel programs are then broken down into several units of execution that can be allocated to different processors and can communicate with each other by means of shared memory.
- Originally parallel systems are considered as those architectures that featured multiple processors sharing the same physical memory and that were considered a single computer.
  - Over time, these restrictions have been relaxed, and parallel systems now include all architectures that are based on the concept of shared memory, whether this is physically present or created with the support of libraries, specific hardware, and a highly efficient networking infrastructure.
  - For example: a cluster of which of the nodes are connected through an InfiniBand network and configured with distributed shared memory system can be considered as a parallel system.
- The term distributed computing encompasses any architecture or system that allows the computation to be broken down into units and executed concurrently on different

computing elements, whether these are processors on different nodes, processors on the same computer, or cores within the same processor.

- Distributed computing includes a wider range of systems and applications than parallel computing and is often considered a more general term.
- Even though it is not a rule, the term distributed often implies that the locations of the computing elements are not the same and such elements might be heterogeneous in terms of hardware and software features.
- Classic examples of distributed computing systems are
  - Computing Grids
  - Internet Computing Systems

Elements of Parallel computing

- Silicon-based processor chips are reaching their physical limits. Processing speed is constrained by the speed of light, and the density of transistors packaged in a processor is constrained by thermodynamics limitations.
- A viable solution to overcome this limitation is to connect multiple processors working in coordination with each other to solve "Grand Challenge" problems.
- The first step in this direction led
  - To the development of parallel computing, which encompasses techniques, architectures, and systems for performing multiple activities in parallel.

a.Parallel Processing

- Processing of multiple tasks simultaneously on multiple processors is called ***parallel processing***.
- The parallel program consists of multiple active processes (tasks) simultaneously solving a given problem.
- A given task is divided into multiple subtasks using a divide-and-conquer technique, and each subtask is processed on a different central processing unit (CPU).
- Programming on multi processor system using the divide-and-conquer technique is called ***parallel programming***.
- Many applications today require more computing power than a traditional sequential computer can offer.

- Parallel Processing provides a cost effective solution to this problem by increasing the number of CPUs in a computer and by adding an efficient communication system between them.
- The workload can then be shared between different processors. This setup results in higher computing power and performance than a single processor a system offers.

Parallel Processing influencing factors

- The development of parallel processing is being influenced by many factors. The prominent among them include the following:
  - Computational requirements are ever increasing in the areas of both scientific and business computing. The technical computing problems, which require high-speed computational power, are related to
    - life sciences, aerospace, geographical information systems, mechanical design and analysis etc.
  - Sequential architectures are reaching mechanical physical limitations as they are constrained by the speed of light and thermodynamics laws.
    - The speed which sequential CPUs can operated is reaching saturation point (no more vertical growth), and hence an alternative way to get high computation speed is to connect multiple CPUs (opportunity for horizontal growth).
  - Hardware improvements in pipelining , super scalar, and the like are non scalable and require sophisticated compiler technology.
    - Developing such compiler technology is a difficult task.
  - Vector processing works well for certain kinds of problems. It is suitable mostly for scientific problems (involving lots of matrix operations) and graphical processing.
    - It is not useful for other areas, such as databases.
  - The technology of parallel processing is mature and can be exploited commercially
    - here is already significant R&D work on development tools and environments.
  - Significant development in networking technology is paving the way for

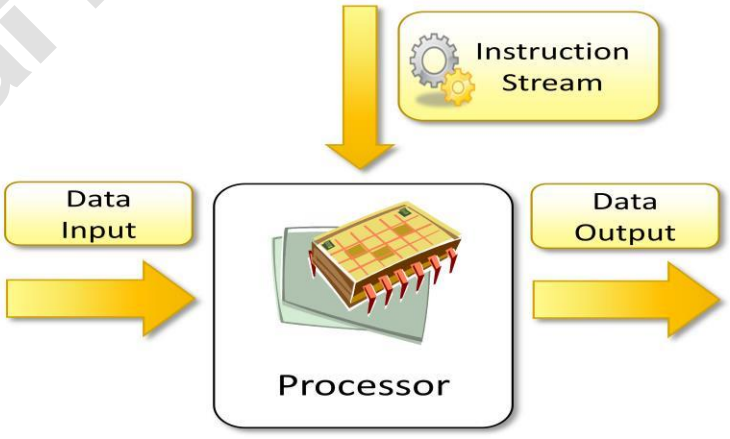
- heterogeneous computing.

b.Hardware architectures for parallel Processing

- The core elements of parallel processing are CPUs. Based on the number of instructions and data streams, that can be processed simultaneously, computing systems are classified into the following four categories:
  - Single-instruction, Single-data (SISD) systems
  - Single-instruction, Multiple-data (SIMD) systems
  - Multiple-instruction, Single-data (MISD) systems
  - Multiple-instruction, Multiple-data (MIMD) systems

(i) Single – Instruction , Single Data (SISD) systems

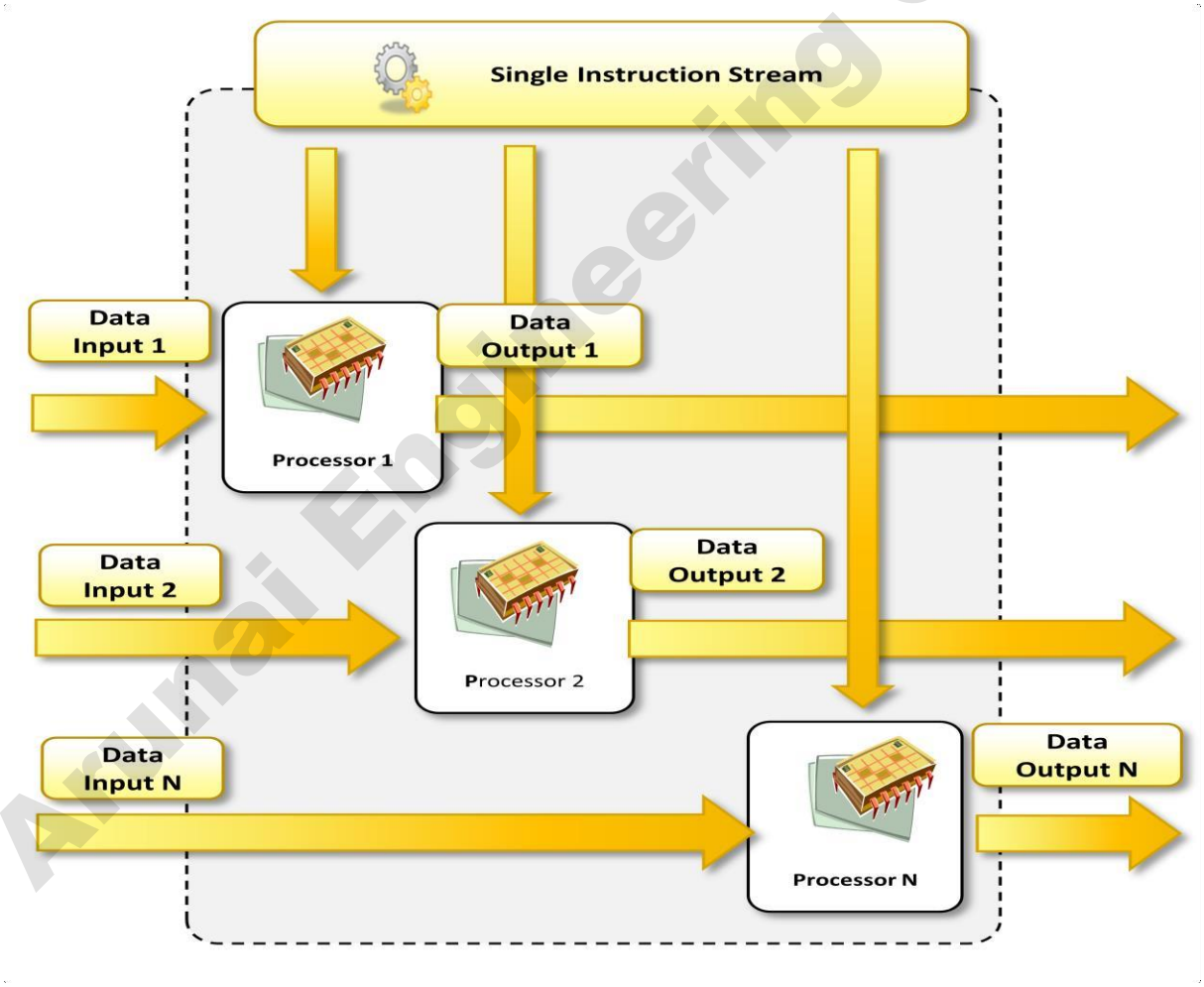
- SISD computing system is a uni-processor machine capable of executing a single instruction, which operates on a single data stream.
- Machine instructions are processed sequentially, hence computers adopting this model are popularly called sequential computers.
- Most conventional computers are built using SISD model.
- All the instructions and data to be processed have to be stored in primary memory.
- The speed of processing element in the SISD model is limited by the rate at which the computer can transfer information internally.
- Dominant representative SISD systems are IBM PC, Macintosh, and workstations.



(ii) Single – Instruction , Multiple Data (SIMD) systems

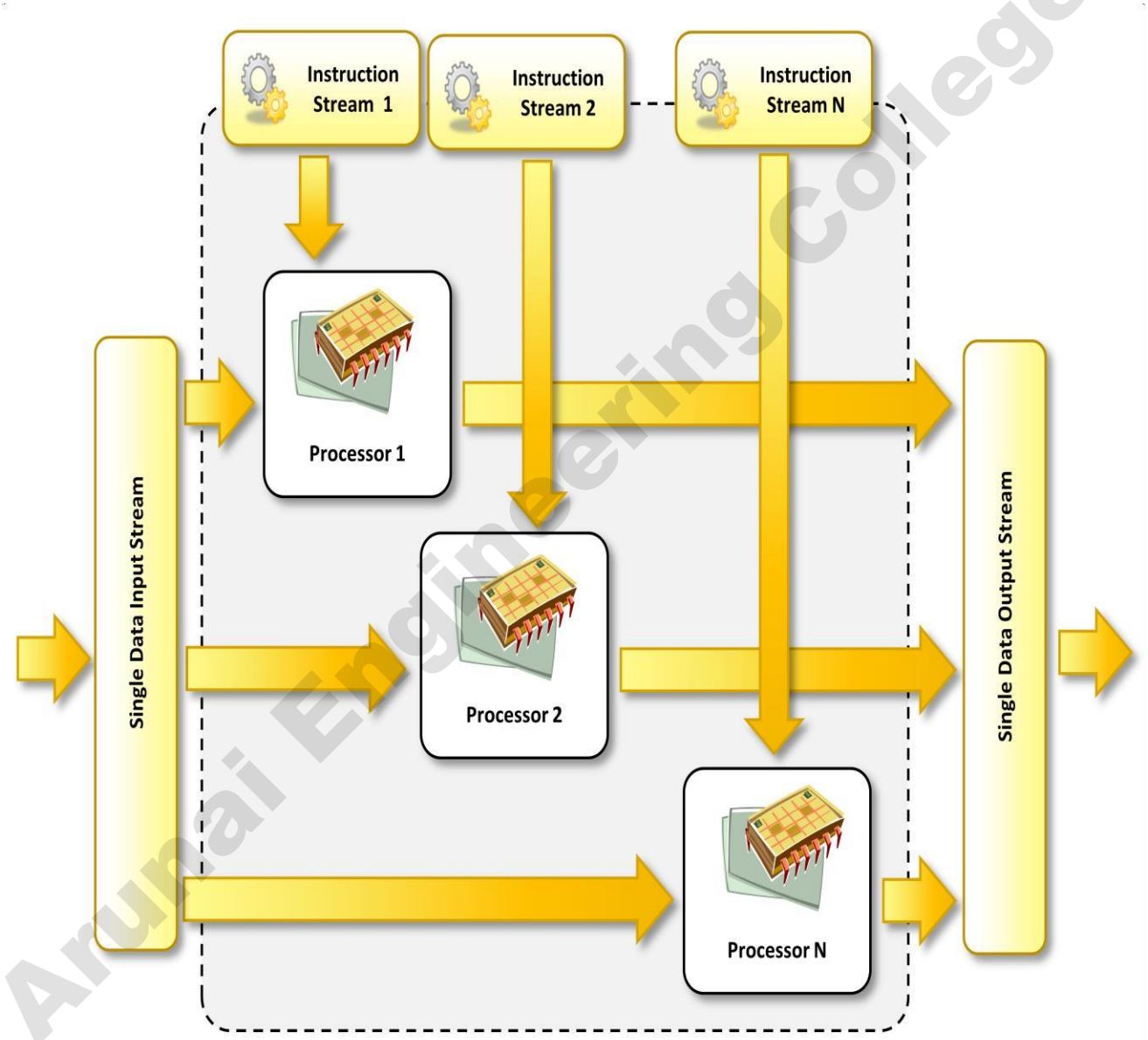
- SIMD computing system is a multiprocessor machine capable of executing the same instruction on all the CPUs but operating on different data streams.
- Machines based on this model are well suited for scientific computing since they involve lots of vector and matrix operations.
- For instance statement Ci = Ai * Bi, can be passed to all the processing elements (PEs), organized data elements of vectors A and B can be divided into multiple sets (N- sets for N PE systems), and each PE can process one data set.

Dominant representative SIMD systems are Cray's Vector processing machine and Thinking Machines Cm*, and GPGPU accelerators



(iii) Multiple – Instruction , Single Data (MISD) systems

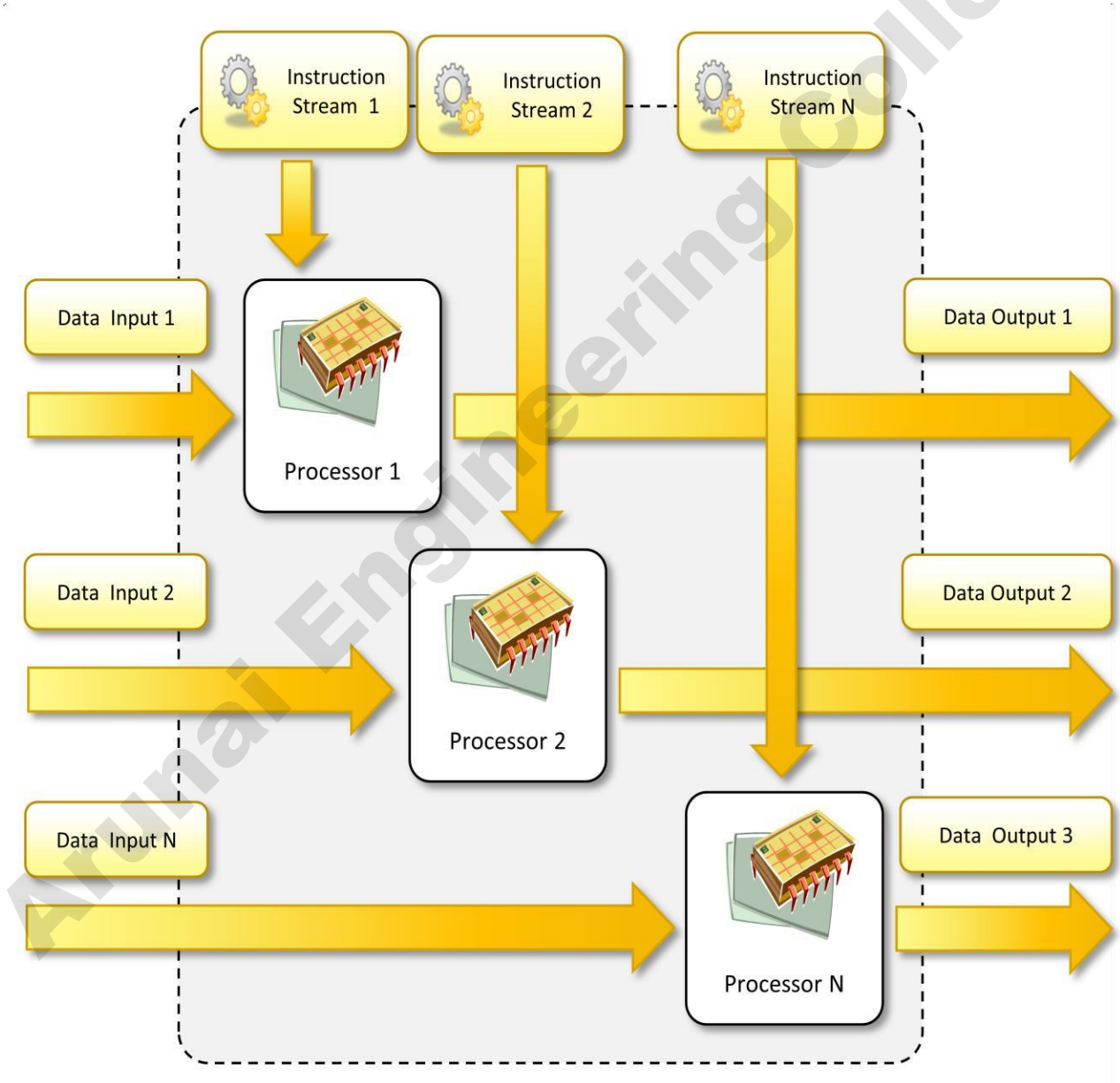
- MISD computing system is a multi processor machine capable of executing different instructions on different Pes all of them operating on the same data set.
- Machines built using MISD model are not useful in most of the applications.
- Few machines are built but none of them available commercially.
- This type of systems are more of an intellectual exercise than a practical configuration.



(iv) Multiple – Instruction , Multiple Data (MIMD) systems

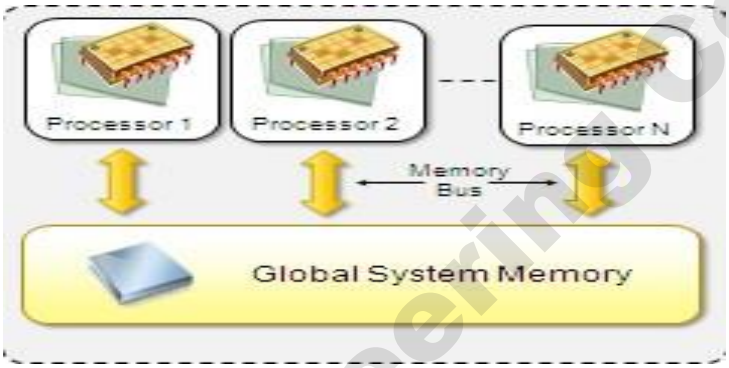
- MIMD computing system is a multi processor machine capable of executing multiple instructions on multiple data sets.
- Each PE in the MIMD model has separate instruction and data streams, hence machines built using this model are well suited to any kind of application.
- Unlike SIMD, MISD machine, PEs in MIMD machines work asynchronously,

MIMD machines are broadly categorized into shared-memory MIMD and distributed memory MIMD based on the way PEs are coupled to the main memory



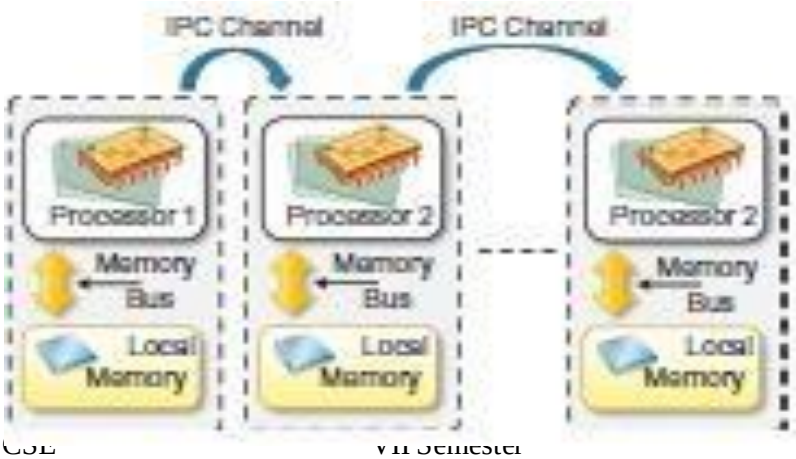
Shared Memory MIMD machines

- All the PEs are connected to a single global memory and they all have access to it.
- Systems based on this model are also called tightly coupled multi processor systems.
- The communication between PEs in this model takes place through the shared memory.
- Modification of the data stored in the global memory by one PE is visible to all other PEs.
- Dominant representative shared memory MIMD systems are silicon graphics machines and Sun/IBM SMP (Symmetric Multi-Processing).



Distributed Memory MIMD machines

- All PEs have a local memory. Systems based on this model are also called loosely coupled multi processor systems.
- The communication between PEs in this model takes place through the interconnection network, the inter process communication channel, or IPC.
- The network connecting PEs can be configured to tree, mesh, cube, and so on.
- Each PE operates asynchronously, and if communication/synchronization among tasks is necessary, they can do so by exchanging messages between them.



Shared Vs Distributed MIMD model

- The shared memory MIMD architecture is easier to program but is less tolerant to failures and harder to extend with respect to the distributed memory MIMD model.
- Failures, in a shared memory MIMD affect the entire system, whereas this is not the case of the distributed model, in which each of the PEs can be easily isolated.
- Moreover, shared memory MIMD architectures are less likely to scale because the addition of more PEs leads to memory contention.
- This is a situation that does not happen in the case of distributed memory, in which each PE has its own memory.

As a result, distributed memory MIMD architectures are most popular today

c.Approaches to Parallel Programming

- A sequential program is one that runs on a single processor and has a single line of control.
- To make many processors collectively work on a single program, the program must be divided into smaller independent chunks so that each processor can work on separate chunks of the problem.
- The program decomposed in this way is a parallel program.
- A wide variety of parallel programming approaches are available.
- The most prominent among them are the following.
- Data Parallelism
- Process Parallelism
- Farmer-and-worker model
- The above said three models are suitable for task-level parallelism. In the case of data level parallelism, the divide-and-conquer technique is used to split data into multiple sets, and each data set is processed on different PEs using the same instruction.
- This approach is highly suitable to processing on machines based on the SIMD model.
- In the case of Process Parallelism, a given operation has multiple (but distinct) activities that can be processed on multiple processors.
- In the case of Farmer-and-Worker model, a job distribution approach is used, one processor is configured as master and all other remaining PEs are designated as slaves,

the master assigns the jobs to slave PEs and, on completion, they inform the master, which in turn collects results.

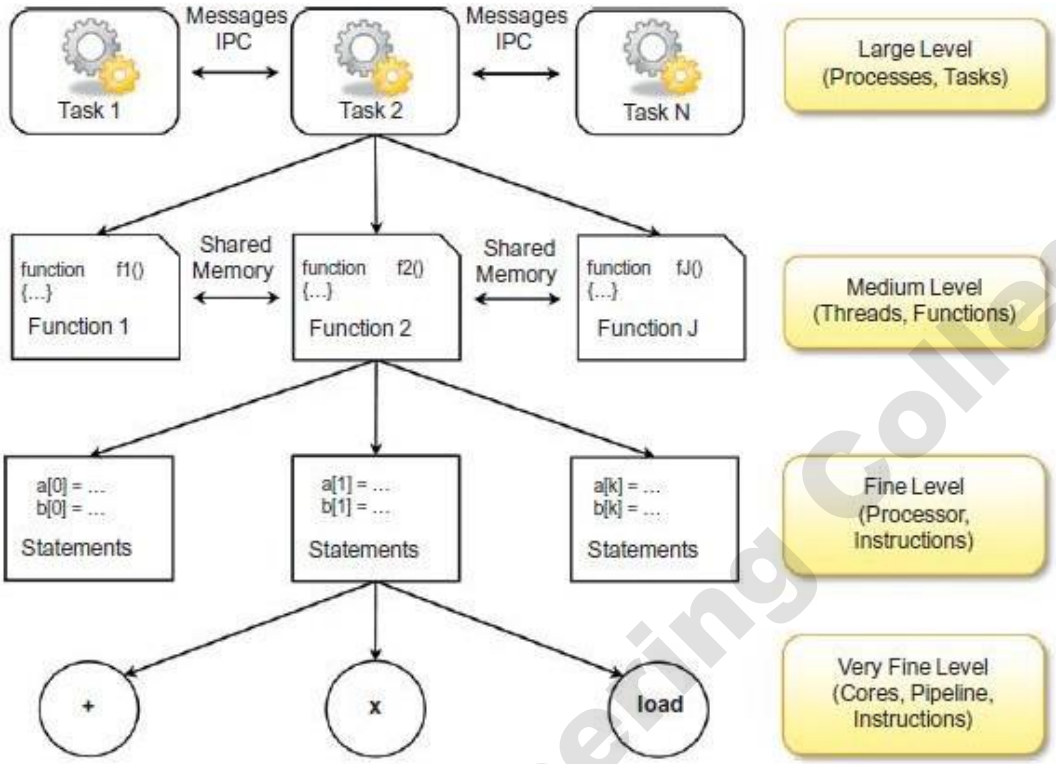
- These approaches can be utilized in different levels of parallelism.

d. Levels of Parallelism

- Levels of Parallelism are decided on the lumps of code (grain size) that can be a potential candidate of parallelism.
- The table shows the levels of parallelism.
- All these approaches have a common goal
  - To boost processor efficiency by hiding latency.
  - To conceal latency, there must be another thread ready to run whenever a lengthy operation occurs.
- The idea is to execute concurrently two or more single-threaded applications. Such as compiling, text formatting, database searching, and device simulation.

Grain Size	Code Item	Parallelized By
Large	Separate and heavy weight process	Programmer
Medium	Function or procedure	Programmer
Fine	Loop or instruction block	Parallelizing compiler
Very Fine	Instruction	Processor

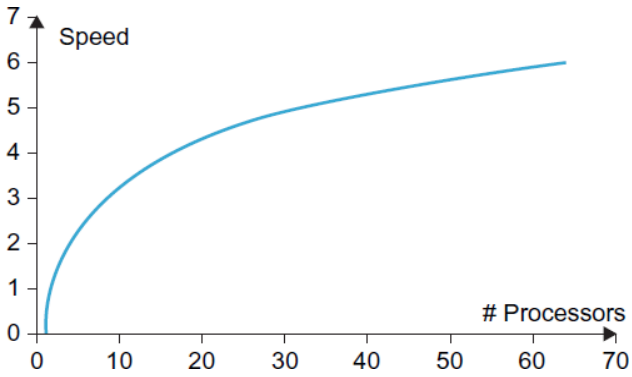
Levels of Parallelism



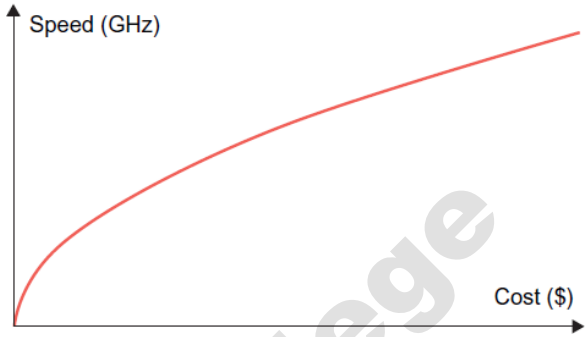
e. Laws of Caution

- Studying how much an application or a software system can gain from parallelism.
- In particular what need to keep in mind is that parallelism is used to perform multiple activities together so that the system can increase its throughput or its speed.
- But the relations that control the increment of speed are not linear.
- For example: for a given n processors, the user expects speed to be increase by in times. This is an ideal situation, but it rarely happens because of the communication overhead.
- Here two important guidelines to take into account.
  - Speed of computation is proportional to the square root of the system cost; they never increase linearly. Therefore, the faster a system becomes, the more expensive it is to increase its speed

– Speed by a parallel computer increases as the logarithm of the number of processors (i.e. y=k*log(N)).



Number processors versus speed



Cost versus speed

Elements of Distributed Computing

a.General concepts and definitions

- Distributed computing studies the models, architectures, and algorithms used for building and managing distributed systems.
- As general definition of the term distributed system, we use the one proposed by Tanenbaum
  - A distributed system is a collection of independent computers that appears to its users as a single coherent system.
- This definition is general enough to include various types of distributed computing systems that are especially focused on unified usage and aggregation of distributed resources.
- Communications is another fundamental aspect of distributed computing. Since distributed systems are composed of more than one computer that collaborate together, it is necessary to provide some sort of data and information exchange between them, which generally occurs through the network.
  - A distributed system is one in which components located at networked computers communicate and coordinate their action only by passing messages.
- As specified in this definition, the components of a distributed system communicate with some sort of message passing. This is a term that encompasses several communication models.

b.Components of distributed System

- A distributed system is the result of the interaction of several components that traverse the entire computing stack from hardware to software.
- It emerges from the collaboration of several elements that- by working together- give users the illusion of a single coherent system.
- The figure provides an overview of the different layers that are involved in providing the services of a distributed system.

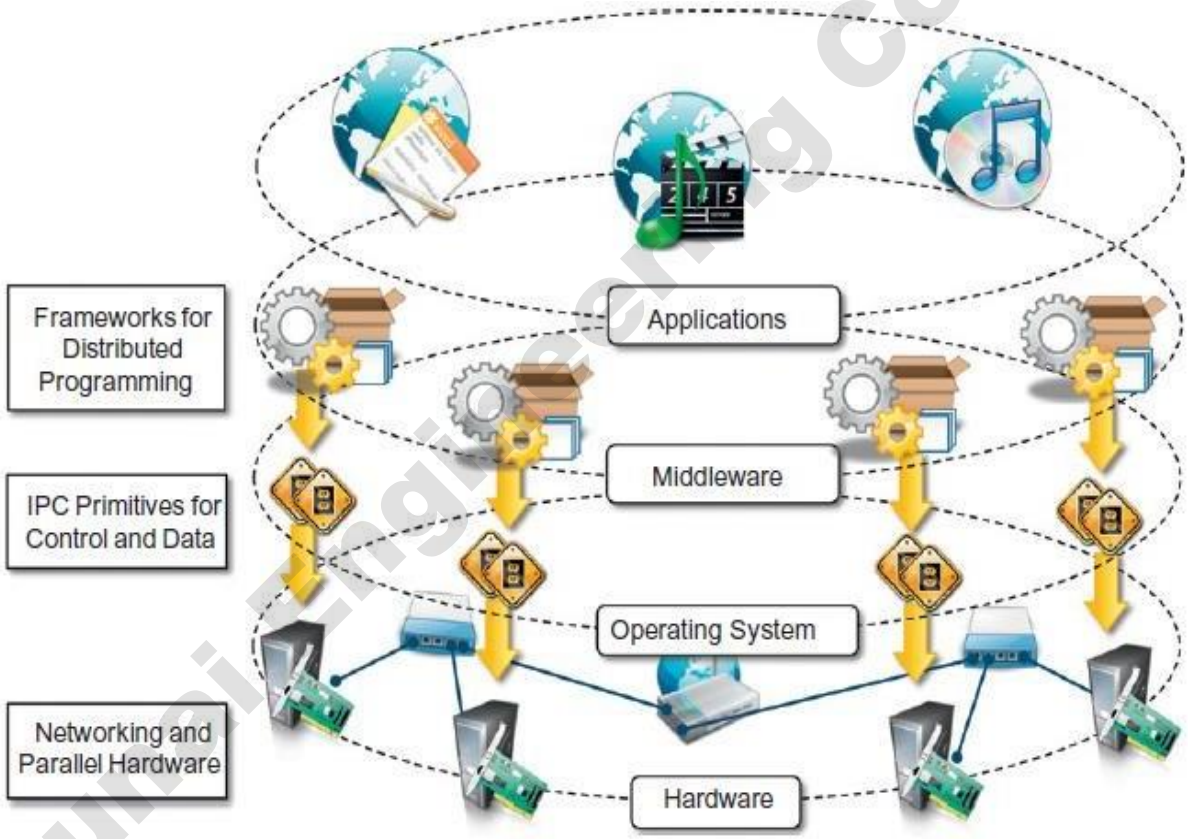


Figure 1.10 A layered view of a distributed system.

c.Architectural styles for distributed computing

- At the very bottom layer, computer and network hardware constitute the physical infrastructure; these components are directly managed by the operating system, which

provides the basic services for inter process communication (IPC), process scheduling and management, and resource management in terms of file system and local devices.

- Taken together these two layers become the platform on top of which specialized software is deployed to turn a set of networked computers into a distributed system
- Although a distributed system comprises the interaction of several layers, the middleware layer is the one that enables distributed computing, because it provides a coherent and uniform runtime environment for applications.
- There are many different ways to organize the components that, taken together, constitute such an environment.
- The interactions among these components and their responsibilities give structure to the middleware and characterize its type or, in other words, define its architecture.
- Architectural styles aid in understanding the classifying the organization of the software systems in general and distributed computing in particular.
- The use of well-known standards at the operating system level and even more at the hardware and network levels allows easy harnessing of heterogeneous components and their organization into a coherent and uniform system.
- For example; network connectivity between different devices is controlled by standards, which allow them into interact seamlessly.
- Design patterns help in creating a common knowledge within the community of software engineers and developers as to how to structure the relevant of components within an application and understand the internal organization of software applications.
- Architectural styles do the same for the overall architecture of software systems.
- The architectural styles are classified into two major classes
    - Software Architectural styles : Relates to the logical organization of the software.
    - System Architectural styles: styles that describe the physical organization of distributed software systems in terms of their major components.

Software Architectural Styles

- Software architectural styles are based on the logical arrangement of software components.
- They are helpful because they provide an intuitive view of the whole system, despite its physical deployment.

- They also identify the main abstractions that are used to shape the components of the system and the expected interaction patterns between them.

Data Centered Architectures

- These architectures **identify the data as the fundamental element of the software system,** and access to shared data is the core characteristics of the data-centered architectures.
- Within the context of distributed and parallel computing systems, **integrity of data is overall goal for such systems.**
- The **repository architectural style** is the most relevant reference model in this category. It is characterized by two main components – the central data structure, which represents the current state of the system, and a collection of independent component, which operate on the central data.
- The ways in which the independent components interact with the central data structure can be very heterogeneous.
- In particular repository based architectures differentiate and specialize further into subcategories according to the choice of control discipline to apply for the shared data structure. Of particular interest are databases and blackboard systems.

Black board Architectural Style

- The black board architectural style is characterized by three main components:
  - Knowledge sources: These are entities that update the knowledge base that is maintained in the black board.
  - Blackboard: This represents the data structure that is shared among the knowledge sources and stores the knowledge base of the application.
  - Control: The control is the collection of triggers and procedures that govern the interaction with the blackboard and update the status of the knowledge base.

Data Flow Architectures

- Access to data is the core feature; data-flow styles explicitly incorporate the pattern of data-flow, since their design is determined by an orderly motion of data from component to component, which is the form of communication between them.
- Styles within this category differ in one of the following ways: how the control is exerted, the degree of concurrency among components, and the topology that describes the flow of data.

- **Batch Sequential:** The batch sequential style is characterized by an ordered sequence of separate programs executing one after the other. These programs are chained together by providing as input for the next program the output generated by the last program after its completion, which is most likely in the form of a file. This design was very popular in the mainframe era of computing and still finds applications today. For example, many distributed applications for scientific computing are defined by jobs expressed as sequence of programs that, for example, pre-filter, analyze, and post process data. It is very common to compose these phases using the batch sequential style.
- **Pipe-and-Filter Style:** It is a variation of the previous style for expressing the activity of a software system as sequence of data transformations. Each component of the processing chain is called a filter, and the connection between one filter and the next is represented by a data stream.

Virtual Machine architectures

- The virtual machine class of architectural styles is characterized by the presence of an abstract execution environment (generally referred as a virtual machine) that simulates features that are not available in the hardware or software.
- Applications and systems are implemented on top of this layer and become portable over different hardware and software environments.
- The general interaction flow for systems implementing this pattern is – the program (or the application) defines its operations and state in an abstract format, which is interpreted by the virtual machine engine. The interpretation of a program constitutes its execution. It is quite common in this scenario that the engine maintains an internal representation of the program state.
- Popular examples within this category are rule based systems, interpreters, and command language processors.
- **Rule-Based Style:**
    - This architecture is characterized by representing the abstract execution environment as an inference engine. Programs are expressed in the form of rules or predicates that hold true. The input data for applications is generally represented by a set of assertions or facts that the inference engine uses to activate rules or to apply predicates, thus transforming data. The examples of rule-based systems can be found in the networking domain: Network Intrusion Detection

Systems (NIDS) often rely on a set of rules to identify abnormal behaviors connected to possible intrusion in computing systems.

- Interpreter Style: The presence of engine to interpret the style.

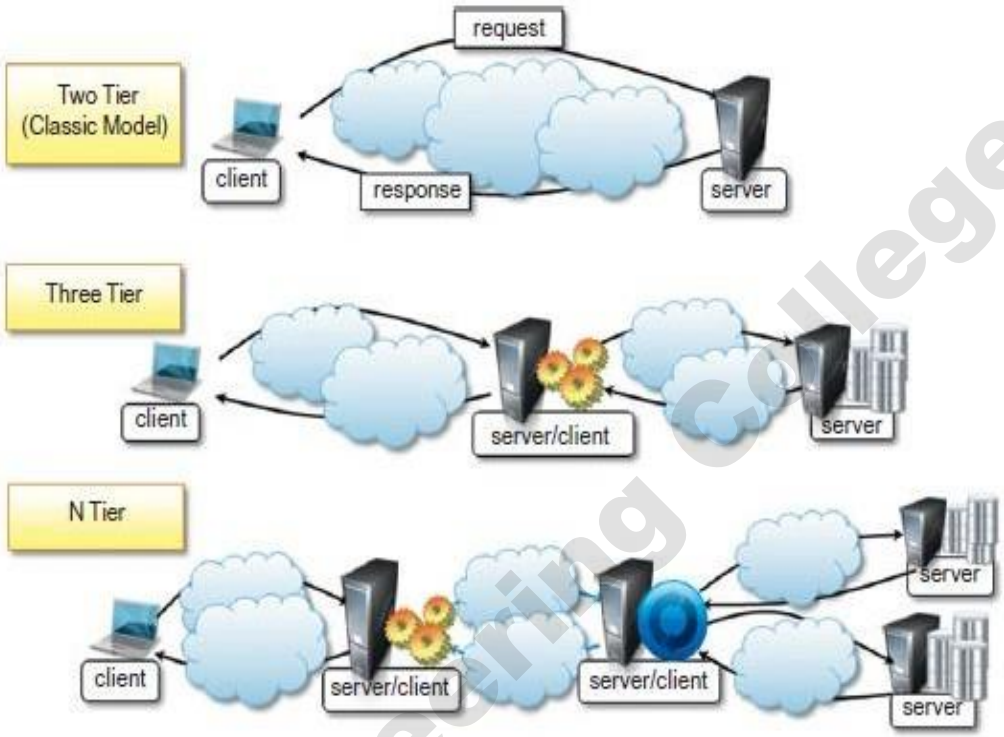
Call and return architectures

- This identifies all systems that are organized into components mostly connected together by method calls.
- The activity of systems modeled in this way is characterized by a chain of method calls whose overall execution and composition identify the execution one or more operations.
- There are three categories in this
  - Top down Style : developed with imperative programming
  - Object Oriented Style: Object programming models
  - Layered Style: provides the implementation in different levels of abstraction of the system.

System Architectural Styles

- System architectural styles cover the physical organization of components and processes over a distributed infrastructure.
- Two fundamental reference style
  - Client / Server
  - Peer- to -Peer
  - The information and the services of interest can be centralized and accessed through a single access point: the server.
  - Multiple clients are interested in such services and the server must be appropriately designed to efficiently serve requests coming from different clients.

Client / Server architectural Styles



- Symmetric architectures in which all the components, called peers, play the same role and incorporate both client and server capabilities of the client/server model.
- More precisely, each peer acts as a server when it processes requests from other peers and as a client when it issues requests to other peers.

Peer-to-Peer architectural Style

d.Models for Inter process Communication

- Distributed systems are composed of a collection of concurrent processes interacting with each other by means of a network connection.
- IPC is a fundamental aspect of distributed systems design and implementation.
- IPC is used to either exchange data and information or coordinate the activity of processes.
- IPC is what ties together the different components of a distributed system, thus making them act as a single system.
- There are several different models in which processes can interact with each other – these maps to different abstractions for IPC.
- Among the most relevant that we can mention are shared memory, remote procedure call (RPC), and message passing.
- At lower level, IPC is realized through the fundamental tools of network programming.
- Sockets are the most popular IPC primitive for implementing communication channels between distributed processes.

Message-based communication

- The abstraction of message has played an important role in the evolution of the model and technologies enabling distributed computing.
- The definition of distributed computing – is the one in which components located at networked computers communicate and coordinate their actions only by passing messages. The term messages, in this case, identify any discrete amount of information that is passed from one entity to another. It encompasses any form of data representation that is limited in size and time, whereas this is an invocation to a remote procedure or a serialized object instance or a generic message.
- The term message-based communication model can be used to refer to any model for IPC.
- Several distributed programming paradigms eventually use message-based communication despite the abstractions that are presented to developers for programming the interactions of distributed components.
- Here are some of the most popular and important:

Message Passing: This paradigm introduces the concept of a message as the main abstraction of the model. The entities exchanging information explicitly encode in the form of a message the data to be exchanged. The structure and the content of a message vary according to the model. Examples of this model are the Message-Passing-Interface (MPI) and openMP.

- **Remote Procedure Call (RPC):** This paradigm extends the concept of procedure call beyond the boundaries of a single process, thus triggering the execution of code in remote processes.
- **Distributed Objects:** This is an implementation of the RPC model for the object-oriented paradigm and contextualizes this feature for the remote invocation of methods exposed by objects. Examples of distributed object infrastructures are Common Object Request Broker Architecture (CORBA), Component Object Model (COM, DCOM, and COM+), Java Remote Method Invocation (RMI), and .NET Remoting.
- **Distributed agents and active Objects:** Programming paradigms based on agents and active objects involve by definition the presence of instances, whether they are agents of objects, despite the existence of requests.
- **Web Service**: An implementation of the RPC concept over HTTP; thus allowing the interaction of components that are developed with different technologies. A Web service is exposed as a remote object hosted on a Web Server, and method invocation are transformed in HTTP requests, using specific protocols such as Simple Object Access Protocol (SOAP) or Representational State Transfer (REST).

e. Technologies for distributed computing

- Remote Procedure Call (RPC)
  - RPC is the fundamental abstraction enabling the execution procedures on clients' request.
  - RPC allows extending the concept of a procedure call beyond the boundaries of a process and a single memory address space.
  - The called procedure and calling procedure may be on the same system or they may be on different systems..
- Distributed Object Frameworks

- Extend object-oriented programming systems by allowing objects to be distributed across a heterogeneous network and provide facilities so that they can be coherently act as though they were in the same address space.

Web Services

- Web Services are the prominent technology for implementing SOA systems and applications.
- They leverage Internet technologies and standards for building distributed systems.
- Several aspects make Web Services the technology of choice for SOA.
- First, they allow for interoperability across different platforms and programming languages.
- Second, they are based on well-known and vendor-independent standards such as HTTP, SOAP, and WSDL.
- Third, they provide an intuitive and simple way to connect heterogeneous software systems, enabling quick composition of services in distributed environment.

ELASTICITY IN CLOUD COMPUTING

- Elasticity is defined as the ability of a system to add and remove resources (such as CPU cores, memory, VM and container instances) to adapt to the load variation in real time.
- Elasticity is a dynamic property for cloud computing.
- Elasticity is the degree to which a system is able to adapt to workload changes by provisioning and deprovisioning resources in an autonomic manner, such that at each point in time the available resources match the current demand as closely as possible.

Elasticity = Scalability + Automation + Optimization

- Elasticity is built on top of scalability.
- It can be considered as an automation of the concept of scalability and aims to optimize at best and as quickly as possible the resources at a given time.
- Another term associated with elasticity is the efficiency, which characterizes how cloud resource can be efficiently utilized as it scales up or down.
- It is the amount of resources consumed for processing a given amount of work, the lower this amount is, the higher the efficiency of a system.

- Elasticity also introduces a new important factor, which is the speed.
- Rapid provisioning and deprovisioning are key to maintaining an acceptable performance in the context of cloud computing
- Quality of service is subjected to a service level agreement

Classification

Elasticity solutions can be arranged in different classes based on

- Scope
- Policy
- Purpose
- Method

a.Scope

- Elasticity can be implemented on any of the cloud layers.
- Most commonly, elasticity is achieved on the IaaS level, where the resources to be provisioned are virtual machine instances.
- Other infrastructure services can also be scaled
- On the PaaS level, elasticity consists in scaling containers or databases for instance.
- Finally, both PaaS and IaaS elasticity can be used to implement elastic applications, be it for private use or in order to be provided as a SaaS
- The elasticity actions can be applied either at the infrastructure or application/platform level.
- The elasticity actions perform the decisions made by the elasticity strategy or management system to scale the resources.
- Google App Engine and Azure elastic pool are examples of elastic Platform as a Service (PaaS).
- Elasticity actions can be performed at the infrastructure level where the elasticity controller monitors the system and takes decisions.
- The cloud infrastructures are based on the virtualization technology, which can be VMs or containers.
- In the embedded elasticity, elastic applications are able to adjust their own resources according to runtime requirements or due to changes in the execution flow.
- There must be a knowledge of the source code of the applications.

- Application Map: The elasticity controller must have a complete map of the application components and instances.
- Code embedded: The elasticity controller is embedded in the application source code.
- The elasticity actions are performed by the application itself.
- While moving the elasticity controller to the application source code eliminates the use of monitoring systems
- There must be a specialized controller for each application.

b.Policy

- Elastic solutions can be either manual or automatic.
- A manual elastic solution would provide their users with tools to monitor their systems and add or remove resources but leaves the scaling decision to them.

Automatic mode: All the actions are done automatically, and this could be classified into reactive and proactive modes.

Elastic solutions can be either reactive or predictive

Reactive mode: The elasticity actions are triggered based on certain thresholds or rules, the system reacts to the load (workload or resource utilization) and triggers actions to adapt changes accordingly.

- An elastic solution is reactive when it scales a posteriori, based on a monitored change in the system.
- These are generally implemented by a set of Event-Condition-Action rules.

Proactive mode: This approach implements forecasting techniques, anticipates the future needs and triggers actions based on this anticipation.

- A predictive or proactive elasticity solution uses its knowledge of either recent history or load patterns inferred from longer periods of time in order to predict the upcoming load of the system and scale according to it.

c.Purpose

- An elastic solution can have many purposes.
- The first one to come to mind is naturally performance, in which case the focus should be put on their speed.
- Another purpose for elasticity can also be energy efficiency, where using the minimum amount of resources is the dominating factor.

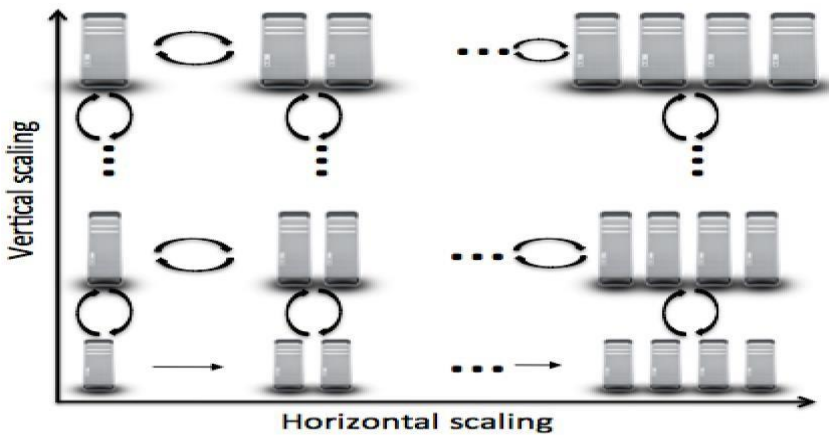
- Other solutions intend to reduce the cost by multiplexing either resource providers or elasticity methods
- Elasticity has different purposes such as improving performance, increasing resource capacity, saving energy, reducing cost and ensuring availability.
- Once we look to the elasticity objectives, there are different perspectives.
- Cloud IaaS providers try to maximize the profit by minimizing the resources while offering a good Quality of Service (QoS),
- PaaS providers seek to minimize the cost they pay to the

Cloud.

- The customers (end-users) search to increase their Quality of Experience (QoE) and to minimize their payments.
- QoE is the degree of delight or annoyance of the user of an application or service

d.Method

- Vertical elasticity, changes the amount of resources linked to existing instances on-the-fly.
- This can be done in two manners.
- The first method consists in explicitly redimensioning a virtual machine instance, i.e., changing the quota of physical resources allocated to it.
- This is however poorly supported by common operating systems as they fail to take into account changes in CPU or memory without rebooting, thus resulting in service interruption.
- The second vertical scaling method involves VM migration: moving a virtual machine instance to another physical machine with a different overall load changes its available resources



- Horizontal scaling is the process of adding/removing instances, which may be located at different locations.
- Load balancers are used to distribute the load among the different instances.
- Vertical scaling **is** the process of modifying resources (CPU, memory, storage or both) size for an instance at run time.
- It gives more flexibility for the cloud systems to cope with the varying workloads

Migration

- Migration can be also considered as a needed action to further allow the vertical scaling when there is no enough resources on the host machine.
- It is also used for other purposes such as migrating a VM to a less loaded physical machine just to guarantee its performance.
- Several types of migration are deployed such as live migration and no-live migration.
- Live migration has two main approaches
    - post-copy
    - pre-copy
- Post-copy migration suspends the migrating VM, copies minimal processor state to the target host, resumes the VM and then begins fetching memory pages from the source.
- In pre-copy approach, the memory pages are copied while the VM is running on the source.
- If some pages are changed (called dirty pages) during the memory copy process, they will be recopied until the number of recopied pages is greater than dirty pages, or the source VM will be stopped.
- The remaining dirty pages will be copied to the destination VM.

Architecture

- The architecture of the elasticity management solutions can be either centralized or decentralized.
- Centralized architecture has only one elasticity controller, i.e., the auto scaling system that provisions and deprovisions resources.

- In decentralized solutions, the architecture is composed of many elasticity controllers or application managers, which are responsible for provisioning resources for different cloud-hosted platforms

Provider

- Elastic solutions can be applied to a single or multiple cloud providers.
- A single cloud provider can be either public or private with one or multiple regions or datacenters.
- Multiple clouds in this context means more than one cloud provider.
- It includes hybrid clouds that can be private or public, in addition to the federated clouds and cloud bursting.
- Most of the elasticity solutions support only a single cloud provider

On-demand Provisioning.

- Resource Provisioning means the selection, deployment, and run-time management of software (e.g., database server management systems, load balancers) and hardware resources (e.g., CPU, storage, and network) for ensuring guaranteed performance for applications.
- Resource Provisioning is an important and challenging problem in the large-scale distributed systems such as Cloud computing environments.
- There are many resource provisioning techniques, both static and dynamic each one having its own advantages and also some challenges.
- These resource provisioning techniques used must meet Quality of Service (QoS) parameters like availability, throughput, response time, security, reliability etc., and thereby avoiding Service Level Agreement (SLA) violation.
- Over provisioning and under provisioning of resources must be avoided.
- Another important constraint is power consumption.
- The ultimate goal of the cloud user is to minimize cost by renting the resources and from the cloud service provider's perspective to maximize profit by efficiently allocating the resources.

- In order to achieve the goal, the cloud user has to request cloud service provider to make a provision for the resources either statically or dynamically.
- So that the cloud service provider will know how many instances of the resources and what resources are required for a particular application.
- By provisioning the resources, the QoS parameters like availability, throughput, security, response time, reliability, performance etc must be achieved without violating SLA

There are two types

- **Static Provisioning**
- **Dynamic Provisioning**

Static Provisioning

- For applications that have predictable and generally unchanging demands/workloads, it is possible to use “static provisioning" effectively.
- With advance provisioning, the customer contracts with the provider for services.
- The provider prepares the appropriate resources in advance of start of service.
- The customer is charged a flat fee or is billed on a monthly basis.

Dynamic Provisioning

- In cases where demand by applications may change or vary, “dynamic provisioning" techniques have been suggested whereby VMs may be migrated on-the-fly to new compute nodes within the cloud.
- The provider allocates more resources as they are needed and removes them when they are not.
- The customer is billed on a pay-per-use basis.
- When dynamic provisioning is used to create a hybrid cloud, it is sometimes referred to as cloud bursting.

Parameters for Resource Provisioning

- Response time
- Minimize Cost
- Revenue Maximization
- Fault tolerant
- Reduced SLA Violation
- Reduced Power Consumption

Response time: The resource provisioning algorithm designed must take minimal time to respond when executing the task.

Minimize Cost: From the Cloud user point of view cost should be minimized.

Revenue Maximization: This is to be achieved from the Cloud Service Provider's view.

Fault tolerant: The algorithm should continue to provide service in spite of failure of nodes.

Reduced SLA Violation: The algorithm designed must be able to reduce SLA violation.

Reduced Power Consumption: VM placement & migration techniques must lower power consumption

Dynamic Provisioning Types

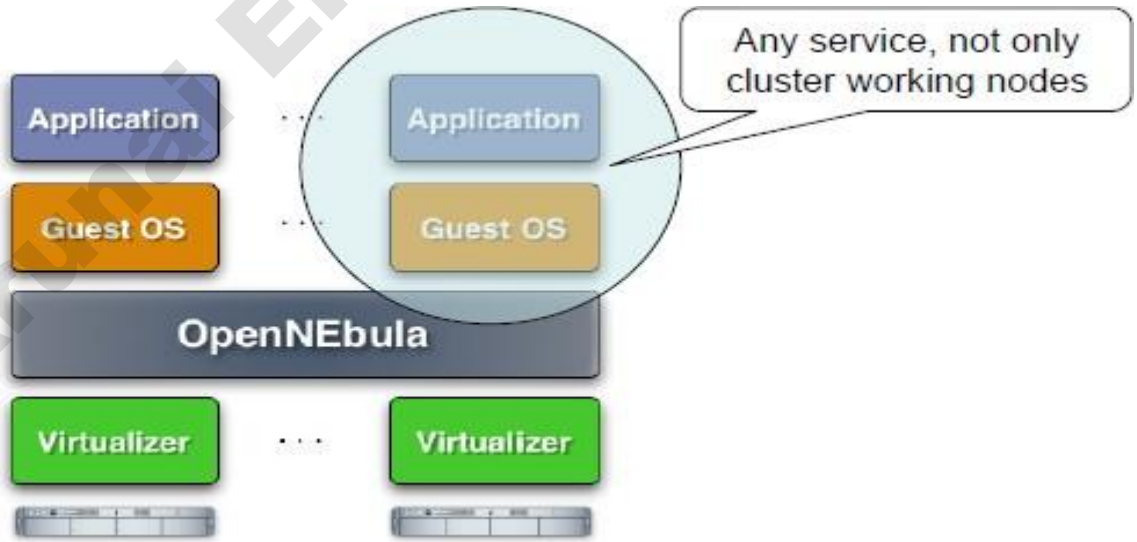
1. Local On-demand Resource Provisioning
2. Remote On-demand Resource Provisioning

Local On-demand Resource Provisioning

1. The Engine for the Virtual Infrastructure

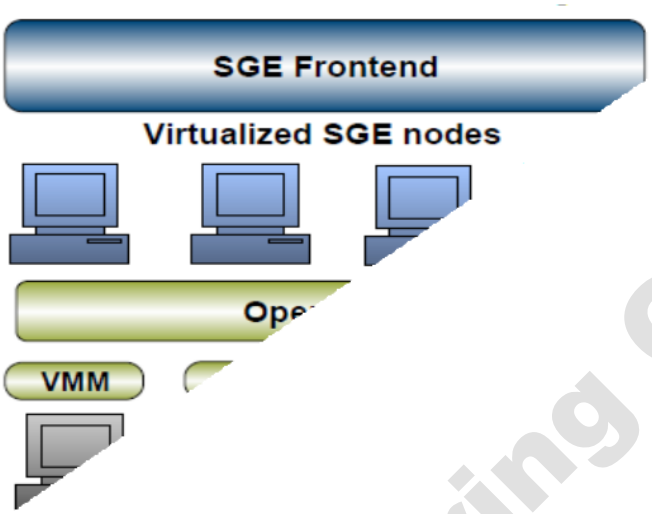
The OpenNebula Virtual Infrastructure Engine

- OpenNEbula creates a distributed virtualization layer
  - Extend the benefits of VM Monitors from one to multiple resources
  - Decouple the VM (service) from the physical location
- Transform a distributed physical infrastructure into a flexible and elastic virtual infrastructure, which adapts to the changing demands of the VM (service) workloads



Separation of Resource Provisioning from Job Management

- New virtualization layer between the service and the infrastructure layers
- Seamless integration with the existing middleware stacks.
- Completely transparent to the computing service and so end users



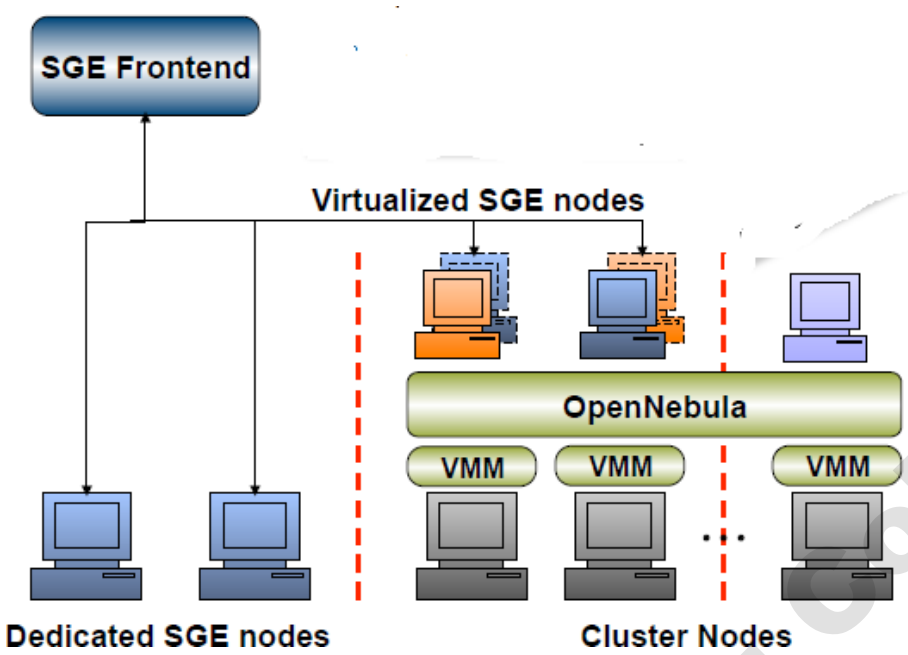
Cluster Partitioning

- Dynamic partition of the infrastructure
- Isolate workloads (several computing clusters)
- Dedicated HA partitions

Benefits for Existing Grid Infrastructures

- The **virtualization of the local infrastructure** supports a virtualized alternative to contribute resources to a Grid infrastructure
    - Simpler deployment and operation of new middleware distributions
    - Lower operational costs
    - Easy provision of resources to more than one infrastructure
    - Easy support for VO-specific worker nodes

  Performance partitioning between local and grid clusters



•

Other Tools for VM Management

- VMware DRS, Platform Orchestrator, IBM Director, Novell ZENworks, Enomalism, Xenoserver
- **Advantages**:
  - Open-source (Apache license v2.0)
  - Open and flexible architecture to integrate new virtualization technologies
  - Support for the definition of any scheduling policy (consolidation, workload balance, affinity, SLA)
  - LRM-like CLI and API for the integration of third-party tools

Remote on-Demand Resource Provisioning

Access to Cloud Systems

- Provision of virtualized resources as a service

VM Management Interfaces

The processes involved are

- Submission
- Control
- Monitoring

Infrastructure Cloud Computing Solutions

- Commercial Cloud: Amazon EC2
- Scientific Cloud: Nimbus (University of Chicago)
- Open-source Technologies
  - Globus VWS (Globus interfaces)
  - Eucalyptus (Interfaces compatible with Amazon EC2)
  - OpenNEbula (Engine for the Virtual Infrastructure)

On-demand Access to Cloud Resources

- Supplement local resources with cloud resources to satisfy peak or fluctuating demands