Unit-I

Introduction to Cloud Computing: Cloud models, Evolution of Cloud Computing, System models for Distributed and Cloud Computing, Cloud Computing Properties and Characteristics, Business Drivers for Adopting Cloud Computing, Software environments for distributed systems and clouds, Performance, Security & Energy Efficiency.

Models of Cloud Computing

Cloud Computing helps in rendering several services according to roles, companies, etc. Cloud computing models are explained below.

- Infrastructure as a service (IaaS)
- Platform as a service (PaaS)
- Software as a service (SaaS)

1. Infrastructure as a service (IaaS)

Infrastructure as a Service (IaaS) helps in delivering computer infrastructure on an external basis for supporting operations. Generally, IaaS provides services to networking equipment, devices, databases, and web servers.

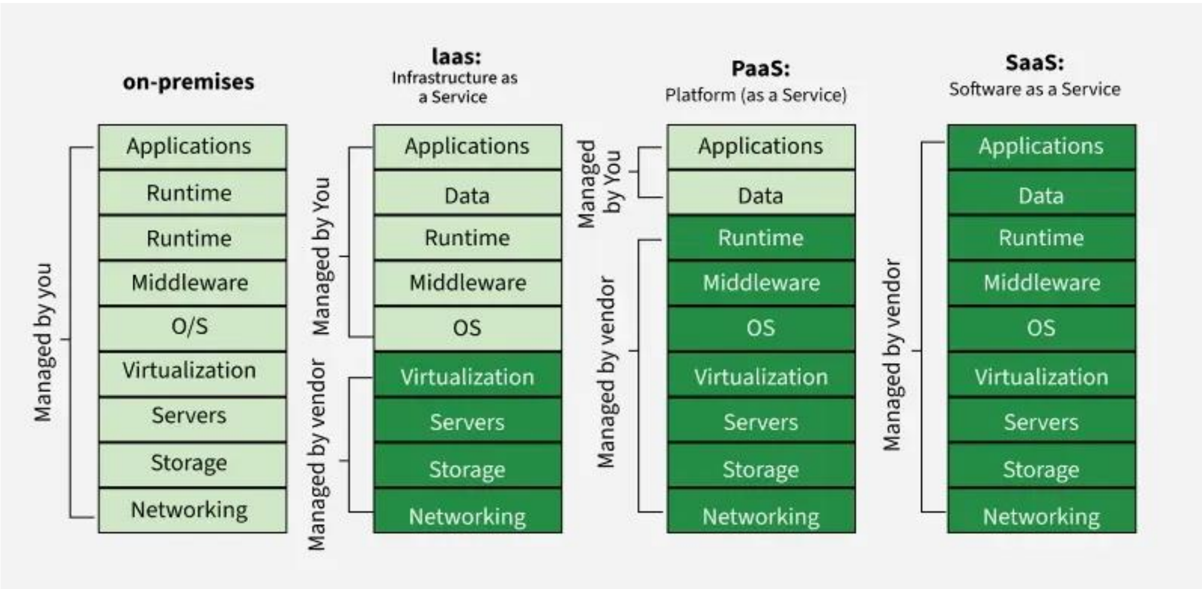
Infrastructure as a Service (IaaS) helps large organizations, and large enterprises in managing and building their IT platforms. This infrastructure is flexible according to the needs of the client.

Advantages of IaaS

- IaaS is cost-effective as it eliminates capital expenses.
- IaaS cloud provider provides better security than any other software.
- IaaS provides remote access.

Disadvantages of IaaS

- In IaaS, users have to secure their own data and applications.
- Cloud computing is not accessible in some regions of the World.



2. Platform as a service (PaaS)

Platform as a Service (PaaS) is a type of cloud computing that helps developers to build applications and services over the Internet by providing them with a platform.

PaaS helps in maintaining control over their business applications.

Advantages of PaaS

- PaaS is simple and very much convenient for the user as it can be accessed via a web browser.
- PaaS has the capabilities to efficiently manage the lifecycle.

Disadvantages of PaaS

- PaaS has limited control over infrastructure as they have less control over the environment and are not able to make some customizations.
- PaaS has a high dependence on the provider.

3. Software as a service (SaaS)

Software as a Service (SaaS) is a type of cloud computing model that is the work of delivering services and applications over the Internet. The SaaS applications are called Web-Based Software or Hosted Software.

SaaS has around 60 percent of cloud solutions and due to this, it is mostly preferred by companies.

Advantages of SaaS

- SaaS can access app data from anywhere on the Internet.
- SaaS provides easy access to features and services.

Disadvantages of SaaS

- SaaS solutions have limited customization, which means they have some restrictions within the platform.
- SaaS has little control over the data of the user.
- SaaS are generally cloud-based, they require a stable internet connection for proper working.

Cloud Deployment Models:

These are the four primary cloud deployment models, categorized by how they are owned and accessed:

Public Cloud: Offered by third-party providers (e.g., AWS, Google Cloud), it's cost-effective but involves shared resources, potentially raising security questions.

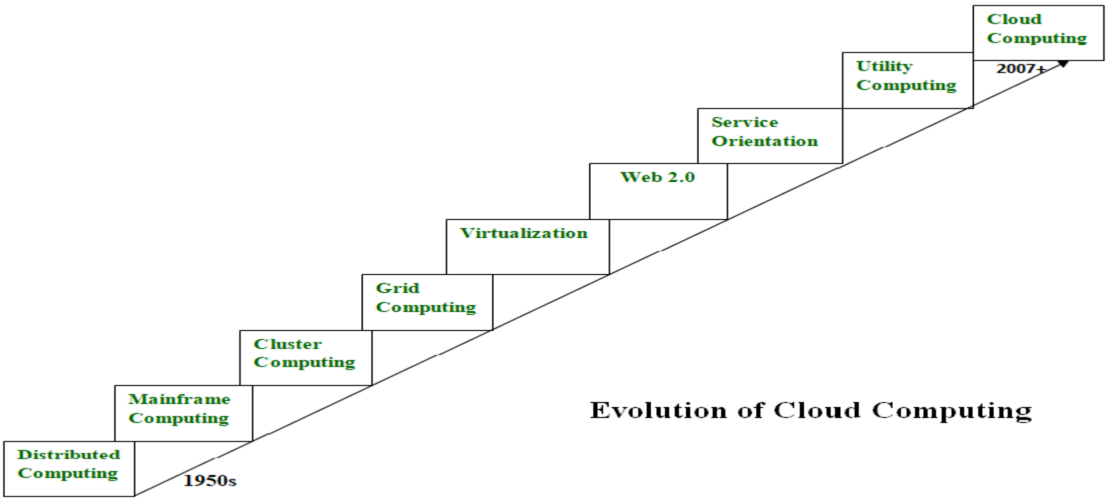
Private Cloud: Dedicated to a single organization, it offers superior security and customization, albeit at a higher cost due to greater control.

Hybrid Cloud: This model integrates public and private clouds, aiming for a balance between cost efficiency and robust security, making it suitable for businesses needing both scalability and data protection.

Community Cloud: A shared infrastructure for multiple organizations with common requirements (e.g., security, compliance), perfect for sectors like healthcare, education, government, and finance.

Evolution of Cloud Computing:

The concept of **Cloud Computing** emerged in the 1950s, evolving from distributed computing to its current form. It enables users to access diverse services like computing resources, storage, applications, and networking via the internet. Major providers include Amazon, Google, and Microsoft.



Distributed Systems (1970-1980):

A Distributed System integrates multiple independent systems, presenting them as a unified entity to users. Its core aim is efficient resource sharing. These systems are characterized by scalability, concurrency, continuous availability, diverse components, and independent failure handling. However, the limitation was the necessity for all systems to be in the same geographical location, which spurred the development of Mainframe, Cluster, and Grid computing to overcome this challenge.

Eg: The World Wide Web (WWW), Online Social Networks, Domain Name System (DNS), Online Gaming, Financial Trading Systems(UPI's), Ride-Sharing Applications(Uber, Ola, Rapido etc).

Mainframe Computing (1950-1970):

Mainframes, introduced in 1951, are powerful, reliable computing machines capable of handling massive data and input/output operations. Still used for bulk processing like online transactions, they boast exceptional uptime and fault tolerance, significantly boosting processing capabilities after distributed computing. However, their high cost led to the development of cluster computing as a more affordable alternative.

Eg: Financial Institutions, Airlines, Government Agencies, Insurance Companies, Retail Chains.

Cluster Computing (1980-1990):

Emerging in the 1980s, cluster computing offered a cost-effective alternative to mainframes. These systems connected multiple machines via high-bandwidth networks, providing comparable computational power and easy scalability through adding new nodes. While addressing the cost issue, clusters still faced the limitation of geographical proximity, a problem that subsequently led to the development of grid computing.

Eg: High-Performance Computing, Web Servers and Application Servers, Database Clusters, Enterprise Resource Planning (ERP) Systems.

Grid Computing (1990-2000):

Introduced in the 1990s, grid computing connected geographically diverse, heterogeneous systems over the internet. While solving location issues, it faced challenges with limited high-bandwidth connectivity due to increased distances. Consequently, cloud computing is often seen as its direct successor, overcoming these network limitations and evolving the concept of distributed resource sharing.

Eg: SETI@home, Large Hadron Collider (LHC) Computing Grid (LCG), Folding@home, Earth Simulator (Japan).

Utility Computing (Late 1990-2000):

Utility Computing is a service provisioning model where computing resources, storage, infrastructure, and other services are delivered and billed based on actual usage, similar to a utility service.

Eg: Amazon Web Services (AWS) EC2, Google Cloud Storage, Microsoft Azure Functions.

Virtualization(1980-Present):

Virtualization, a 40-year-old concept, creates a virtual layer enabling multiple software instances on single hardware. It's a foundational technology for cloud computing services like Amazon EC2, with hardware virtualization remaining prevalent.

Web 2.0:

Web 2.0, gaining popularity in 2004, serves as the interactive interface between cloud computing services and users. It enables the dynamic, flexible web pages we see today. This technology is crucial for platforms like Google Maps, Facebook, and Twitter, making modern social media experiences possible.

Service Orientation:

Service orientation, a reference model for cloud computing, enables low-cost, flexible, and evolvable applications. It introduced key concepts like Quality of Service (QoS), including Service Level Agreements (SLA), and formalized Software as a Service (SaaS), shaping modern cloud offerings.

System Models for Distributed Computing:

System models for Distributed and Cloud Computing describe the various architectural paradigms and fundamental structures that define how these systems are built, organized, and interact.

Distributed computing systems are designed to coordinate multiple independent computing entities to appear as a single, coherent system. Key architectural models include:

1. **Client-Server Model:**
    - **Description:** The most common model. Clients request services (e.g., data, resources, computation) from servers. Servers respond to these requests.
    - **Characteristics:** Centralized control (at the server), clear separation of concerns.
    - **Examples:** Web browsers (client) accessing websites (server), email clients accessing mail servers, file servers.

2. **Peer-to-Peer (P2P) Model:**
    - **Description:** All participants (peers) can act as both clients and servers, directly interacting with each other. There is no central authority managing all communication.
    - **Characteristics:** Decentralized, fault-tolerant (no single point of failure if peers go offline), scalable, resource sharing.
    - **Examples:** File-sharing networks (BitTorrent), some cryptocurrencies (e.g., Bitcoin blockchain), distributed content delivery.
3. **Layered Model (Tiered Architecture):**
    - **Description:** Components are organized into distinct layers, where each layer provides services to the layer above it and requests services from the layer below. Common layers include presentation (user interface), application logic, and data storage.
    - **Characteristics:** Modularity, easier management, scalability of individual layers.
    - **Examples:** 3-tier architecture (presentation, application, data), network protocols (OSI model).
4. **Object-Based Model / Distributed Objects:**
    - **Description:** Systems are composed of distributed objects that interact by invoking methods on each other. The location of the object is transparent to the caller.
    - **Characteristics:** Encapsulation, reusability, location transparency.
    - **Examples:** CORBA (Common Object Request Broker Architecture), Java RMI (Remote Method Invocation), DCOM (Distributed Component Object Model).
5. **Message Passing Model:**
    - **Description:** Components communicate by sending messages to each other. This is a fundamental underlying mechanism for many other distributed models.
    - **Characteristics:** Decoupling of senders and receivers, asynchronous communication.
    - **Examples:** Actors model, Erlang, Kafka message queues.

System Models for Cloud Computing

Cloud computing builds upon distributed system principles, abstracting the underlying complexity and offering resources as services. The system models here often relate to how resources are managed, provisioned, and consumed.

1. **Resource Pooling Model:**
    - **Description:** The provider's computing resources (e.g., processors, memory, storage, network bandwidth) are pooled to serve multiple consumers using a multi-tenant model. Resources are dynamically assigned and reassigned according to demand.

- **Characteristics:** Efficiency, elasticity, cost-effectiveness.
- **Relation to Distributed Systems:** Achieved through virtualization and distributed resource management.

2. **Virtualization Model:**
   - **Description:** This is a core underlying technology. It creates virtual versions of hardware components (e.g., virtual machines, virtual networks, virtual storage) allowing multiple isolated instances to run on a single physical machine.
   - **Characteristics:** Isolation, resource multiplexing, portability, foundation for IaaS.
   - **Examples:** VMware, KVM, Xen.
3. **Service Models (IaaS, PaaS, SaaS):**
   - **Description:** These define the layers of abstraction offered to the user.
     - **IaaS:** Infrastructure as a Service (e.g., virtual machines, networks). User manages OS and above.
     - **PaaS:** Platform as a Service (e.g., development environment, databases). User manages applications and data.
     - **SaaS:** Software as a Service (e.g., complete applications like Gmail). User only interacts with the application.
   - **Characteristics:** Varying levels of provider management and user control.
4. **Deployment Models (Public, Private, Hybrid, Community):**
   - **Description:** Define the scope and location of the cloud infrastructure.
     - **Public Cloud:** Shared, internet accessible.
     - **Private Cloud:** Exclusive to one organization.
     - **Hybrid Cloud:** Mix of public and private.
     - **Community Cloud:** Shared among specific organizations with common concerns.
   - **Characteristics:** Impact security, cost, control, and compliance.
5. **Elasticity and Scalability Model:**
   - **Description:** Cloud systems are designed to rapidly provision and de-provision computing capabilities in an elastic manner, scaling resources up or down automatically based on demand.
   - **Characteristics:** On-demand resource allocation, cost optimization, performance optimization.
   - **Relation to Distributed Systems:** Leverages distributed resource management and load balancing.

Cloud Computing Properties and Characteristics:

Cloud computing possesses several defining properties and characteristics, making it a highly flexible and efficient IT delivery model.

Key Characteristics:

1. **On-Demand Self-Service:** Users can provision computing capabilities, such as server time and network storage, automatically without requiring human interaction with each service provider.
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).
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.
4. **Rapid Elasticity:** Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand. To the consumer, the capabilities available for provisioning often appear unlimited.
5. **Measured Service:** Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth). Usage is monitored, controlled, and reported, providing transparency for both the provider and consumer.

Business Drivers for Adopting Cloud Computing:

Cloud computing is widely adopted due to its numerous business benefits, which help shift IT's role from a cost burden to a strategic advantage.

1. **Cost Optimization:** Shift from large upfront IT investments (CapEx) to flexible operational spending (OpEx). The pay-as-you-go model reduces costs by eliminating over-provisioning and allows funds to be reallocated strategically.
2. **Enhanced Agility and Speed to Market:** Cloud enables rapid resource provisioning and application deployment. Businesses can quickly innovate, respond to market changes, and get products to customers faster than traditional IT setups.
3. **Scalability and Elasticity:** Cloud environments instantly scale resources (compute, storage) up or down based on fluctuating demand. This prevents over-provisioning, ensures peak performance, and optimizes resource utilization and costs.
4. **Focus on Core Business:** Offloading IT infrastructure management frees internal staff and resources. Businesses can then concentrate on strategic initiatives, innovation, and core competencies that directly drive revenue and competitive advantage.
5. **Innovation and Access to Advanced Technologies:** Cloud providers continually invest in cutting-edge technologies like AI/ML, IoT, and serverless. Adopting cloud grants immediate access to these tools, enabling faster innovation without significant upfront R&D costs.
6. **Improved Disaster Recovery and Business Continuity:** Cloud's distributed infrastructure offers inherent resilience. Businesses can implement cost-effective

disaster recovery plans, ensuring robust data protection and continuous application availability even during outages.

7. **Global Reach and Performance:** Cloud's worldwide data centers allow deploying applications closer to global users. This significantly reduces latency, improves user experience, and facilitates seamless expansion into new international markets.

8. **Enhanced Collaboration and Remote Work:** Cloud-based applications and services facilitate seamless teamwork among dispersed employees. They provide ubiquitous, secure access to necessary tools and data, effectively enabling modern remote and hybrid work models.

9. **Security and Compliance:** Cloud providers invest heavily in advanced security measures, certifications (e.g., GDPR, HIPAA), and expert teams. This often offers a higher level of security and easier compliance adherence than many businesses can achieve on-premises.

Software environments for distributed systems and clouds:

1. Distributed Systems Software Environments

Distributed systems consist of multiple independent computers working together as a single system. Key software environments include:

a) **Middleware**

- Acts as a bridge between applications and operating systems.
- Examples: CORBA, Java RMI, Microsoft COM+, Apache Thrift.

b) **Distributed Operating Systems**

- Manage hardware and software resources across networked computers.
- Examples: Amoeba, Plan 9, V-System.

c) **Communication Protocols**

- Support reliable data exchange.
- Examples: TCP/IP, gRPC, MPI (Message Passing Interface).

d) **Cluster Management Software**

- For managing clusters of machines.
- Examples: Kubernetes (can also be used in cloud), Apache Mesos, SLURM.

2. Cloud Computing Software Environments

Cloud environments offer scalable, on-demand computing services over the internet. Software platforms here include:

a) Virtualization Platforms

- Enable multiple OS instances on a single machine.
- **Examples**: VMware, Hyper-V, KVM.

b) Cloud Operating Systems/Orchestration Tools

- Manage cloud resources and deployment.
- **Examples**: OpenStack, Kubernetes, Cloud Foundry.

c) Platform-as-a-Service (PaaS)

- Provides runtime environments for app development.
- **Examples**: Google App Engine, Heroku, Microsoft Azure App Services.

d) Infrastructure-as-a-Service (IaaS) Tools

- Offer virtualized computing resources.
- **Examples**: Amazon EC2, Microsoft Azure VM, Google Compute Engine.

e) Monitoring & Management Tools

- For performance, availability, and cost tracking.
- **Examples**: Prometheus, Grafana, Datadog, AWS CloudWatch.

3. Common Features in Both Environments

- **Scalability**: Automatic resource adjustment based on load.
- **Fault Tolerance**: Redundancy and recovery mechanisms.
- **Security**: Encryption, identity management, and access control.
- **APIs & SDKs**: For integration and automation.

Performance in Cloud Computing:

a) Key Factors Affecting Performance

- **Network latency**: Delay in data transmission due to geographic or bandwidth limits.
- **Resource allocation**: Inefficient VM or container allocation can cause bottlenecks.
- **Scalability**: Auto-scaling may lag if not configured properly.
- **Load balancing**: Poor load distribution can degrade service responsiveness.

b) Performance Optimization Techniques

- Auto-scaling and load balancing (e.g., AWS Auto Scaling, Azure Load Balancer)
- Caching mechanisms (e.g., Redis, CDN)
- Efficient VM/container placement using orchestration tools (e.g., Kubernetes).

Security in Cloud Computing

a) Security Challenges

- **Data breaches** and unauthorized access
- **Multi-tenancy risks**: Shared environments can expose sensitive data.
- **Insecure APIs**: Poorly designed interfaces can be exploited.

- **Insider threats**: Employees with elevated access may compromise systems.

b) Security Measures

- **Encryption** (in transit and at rest)
- **Identity and Access Management (IAM)** (e.g., AWS IAM, Azure AD)
- **Firewalls and Intrusion Detection Systems**
- **Regular audits and compliance** (ISO 27001, GDPR, HIPAA)

Energy Efficiency in Cloud Computing

a) Why It Matters

- Data centres consume large amounts of electricity.
- Energy-efficient cloud systems reduce operational costs and carbon footprint.

b) Techniques for Energy Efficiency

- **Virtualization**: Improves resource utilization, reducing hardware needs.
- **Dynamic voltage and frequency scaling (DVFS)**: Adjusts power use based on workload.
- **Workload consolidation**: Shifts tasks to fewer active servers.
- **Green data centres**: Use of renewable energy, advanced cooling systems.

Aspect	Key Issues	Solutions / Best Practices
Performance	Latency, scalability, load balancing	Auto-scaling, caching, optimized placement.
Security	Data breaches, API risks, insider threats	Encryption, IAM, firewalls, compliance
Energy	High power consumption, heat generation	Virtualization, DVFS, green infrastructure