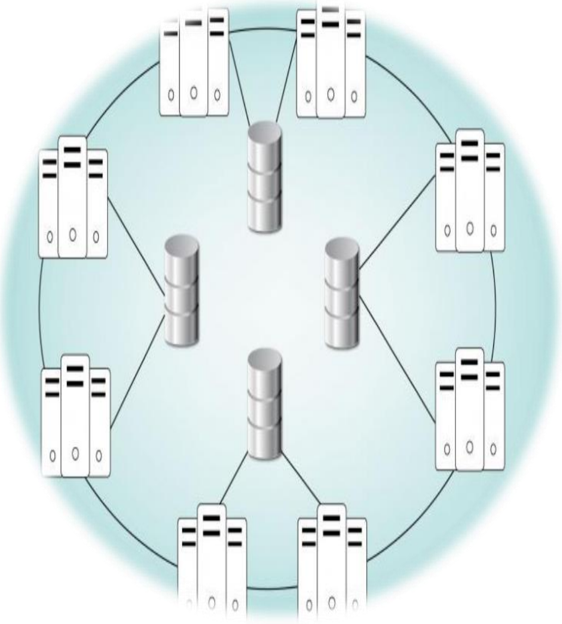
Information Technology Institute

ITI

Introduction to Computer Networks & Cyber Security

Prepared By : Mohamed AboSehly

Distributed Systems & Cloud Computing



Session 3 (Distributed Systems)

- **Outlines**
  - **Distributed Systems overview**
    - Definition and Basic Terminologies
  - **Why build a distributed system?**
  - **Types of Distributed Systems**
    - The 4 Distributed Systems architecture
  - **Distributed System Examples**
  - **Cloud computing**
    - Cloud computing service models
    - Cloud computing deployment models

What is Distributed Systems ?

Distributed Systems

-Is a group of computers working together as to appear as a single computer to the end-user.

-Is a collection of independent components located on different machines that share messages with each other in order to **achieve common goals.**

Centralized system VS Distributed system

- **Centralized system:** State stored on a **single** computer
  - Simpler
  - Easier to understand
  - Can be faster for a single user

- **Distributed system:** State divided over **multiple** computers
  - More robust (can tolerate failures)
  - More scalable (often supports many users)
  - More complex

Why build a distributed system?

- **One interface** to the end-user.
- **Performance**
  - maximize resources and information while preventing failures
- **Reliability**
  - if one system fails, it won't affect the availability of the service
- **Dependency on cloud**
- **Scaling**

Distributed system is growing…They are everywhere!

- modern applications no longer run in isolation. The vast majority of products and applications rely on distributed systems such as :

- **Networks**
  - LAN/ Internet
- **Distributed Real-time Systems**
  - **Uber and logistics** use real-time tracking systems.
- **Parallel Processing**
  - Could Computing
- **Distributed Database Systems**
  - **Multiple servers and/or physical locations**. The data can either be **replicated o**r duplicated across systems.

Distributed System Architecture

Distributed systems must have :

- **Network that connects all components**
  - Hardware, or Software) together to communicate with each other with an IP address
- **Messages passed** between machines contain forms of data that the systems want to share like databases, objects, and files.

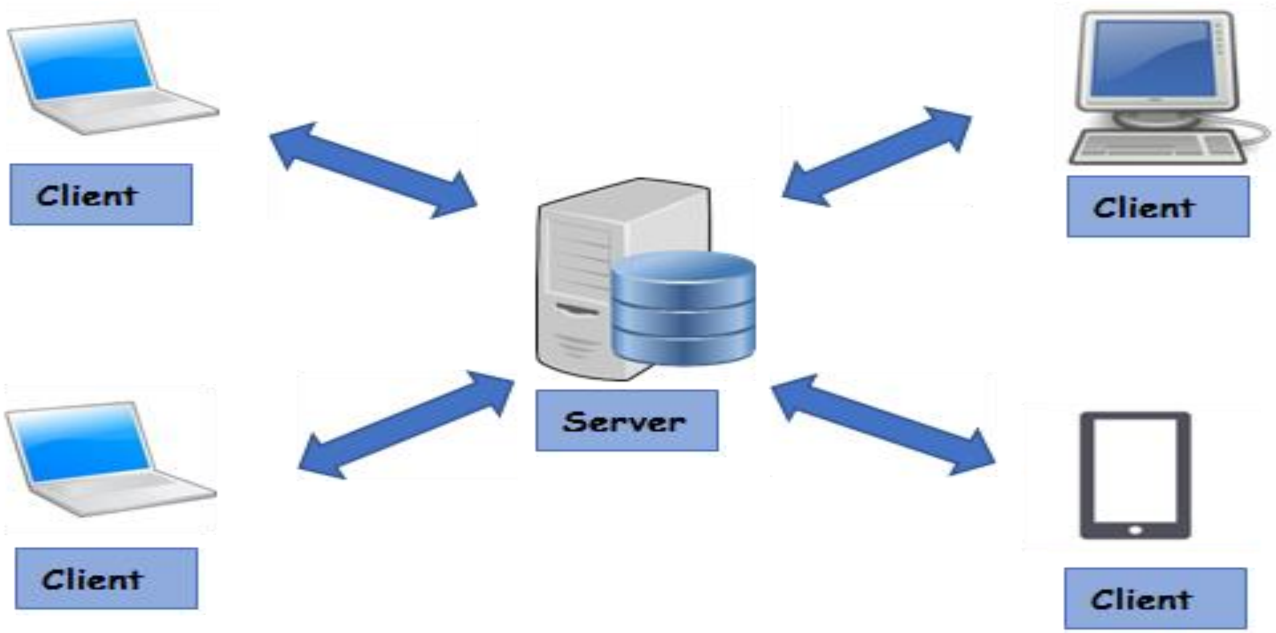
- **The way the messages are communicated** reliably whether it's sent, received, acknowledged or how a node retries on failure is an important feature of a distributed system.

Types of Distributed System Architectures:

Four architecture types :

1- Client-server:

- a server as a shared resource like (a printer, database, or a web server)
- Multiple clients use the shared resource.

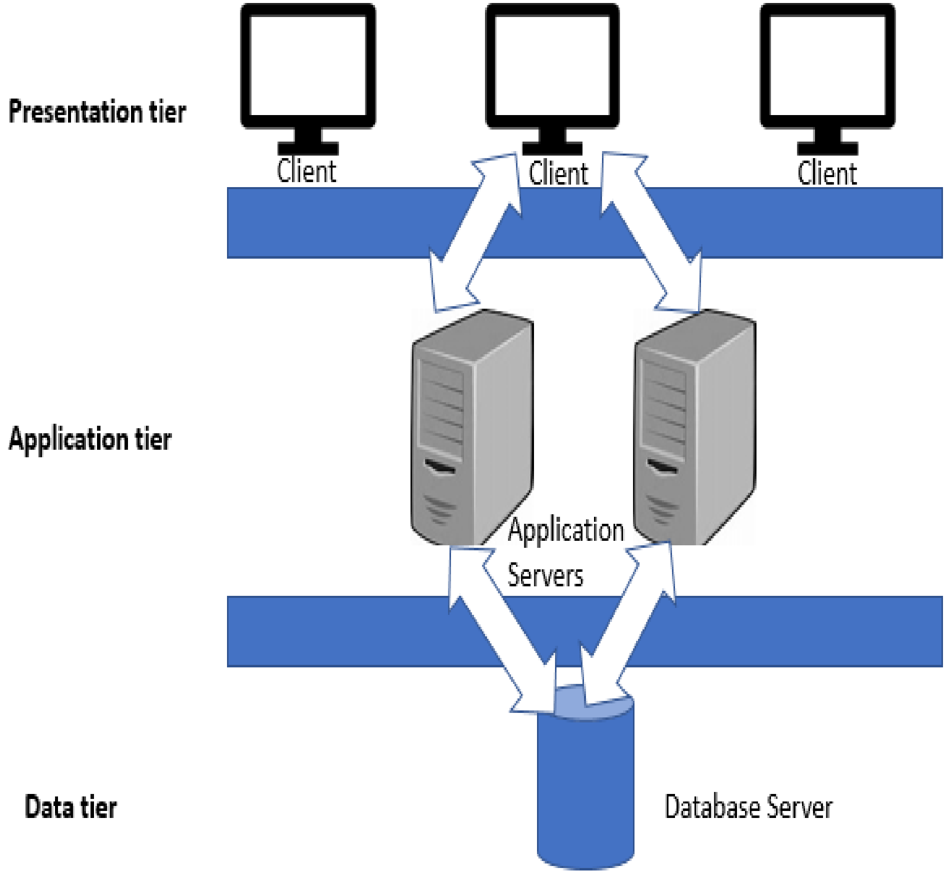


Types of Distributed System Architectures:

Four architecture types :

2- Three-tier:

- clients no longer need to be intelligent
- can rely on a **middle tier** to do the **processing and decision making**.
- Most of the first web applications fall under this category.
- **The middle tier** could be **called an agent** that receives requests from clients, and then forwards it on to the servers.



Types of Distributed System Architectures:

Four architecture types :

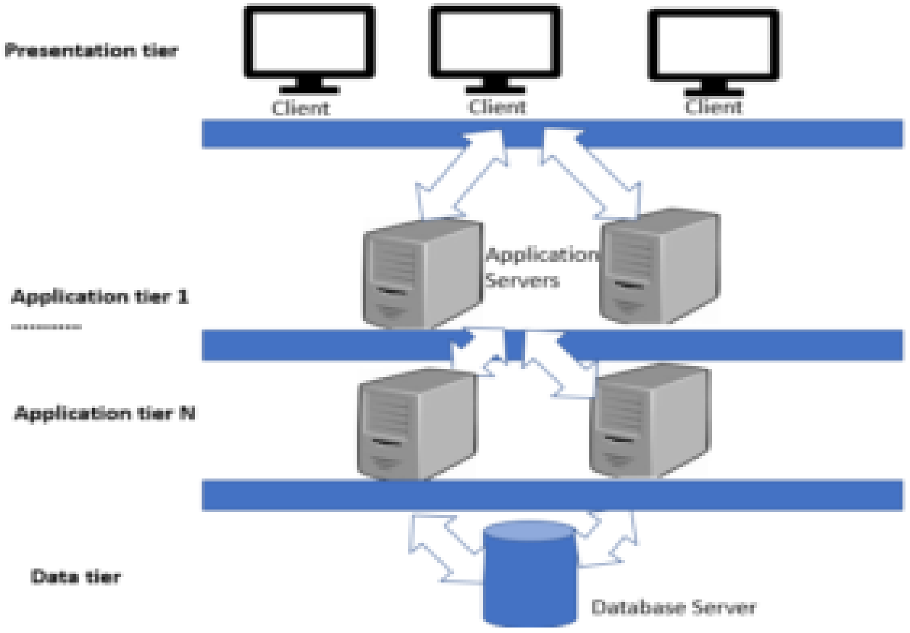
3- Multi-tier:

- Enterprise web services first created **n-tier or multi-tier systems architectures**.

-This popularized the application servers that contain the business logic

- n-tier interacts both with the data tiers and presentation tiers.

- Ex : google.com



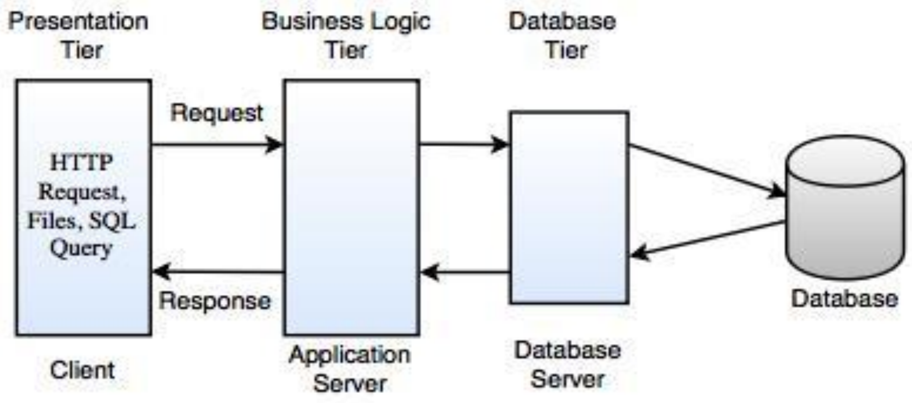


Fig. Multi-Tier Architecture

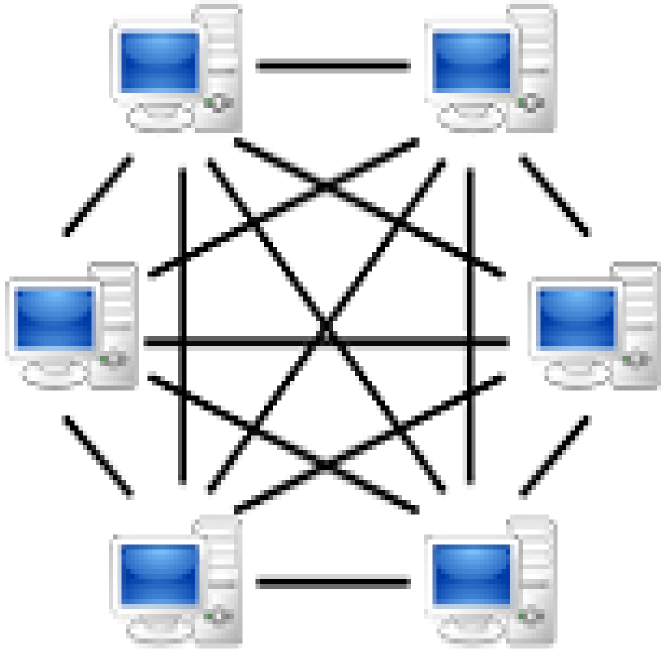
Types of Distributed System Architectures:

Four architecture types :

4- Peer-to-peer :

- **<u>No centralized</u>** or special machine that does the heavy lifting and intelligent work in this architecture.

- All the **decision** making and responsibilities are **<u>split up amongst the machines</u>** involved and each could take on client or server roles.
  - Blockchain is a good example of this.



Distributed System "Common Characteristics"/ most important functions

What are we trying to achieve when we construct a distributed system?

Certain common characteristics can be used to assess distributed systems

1- Heterogeneity

2- Openness

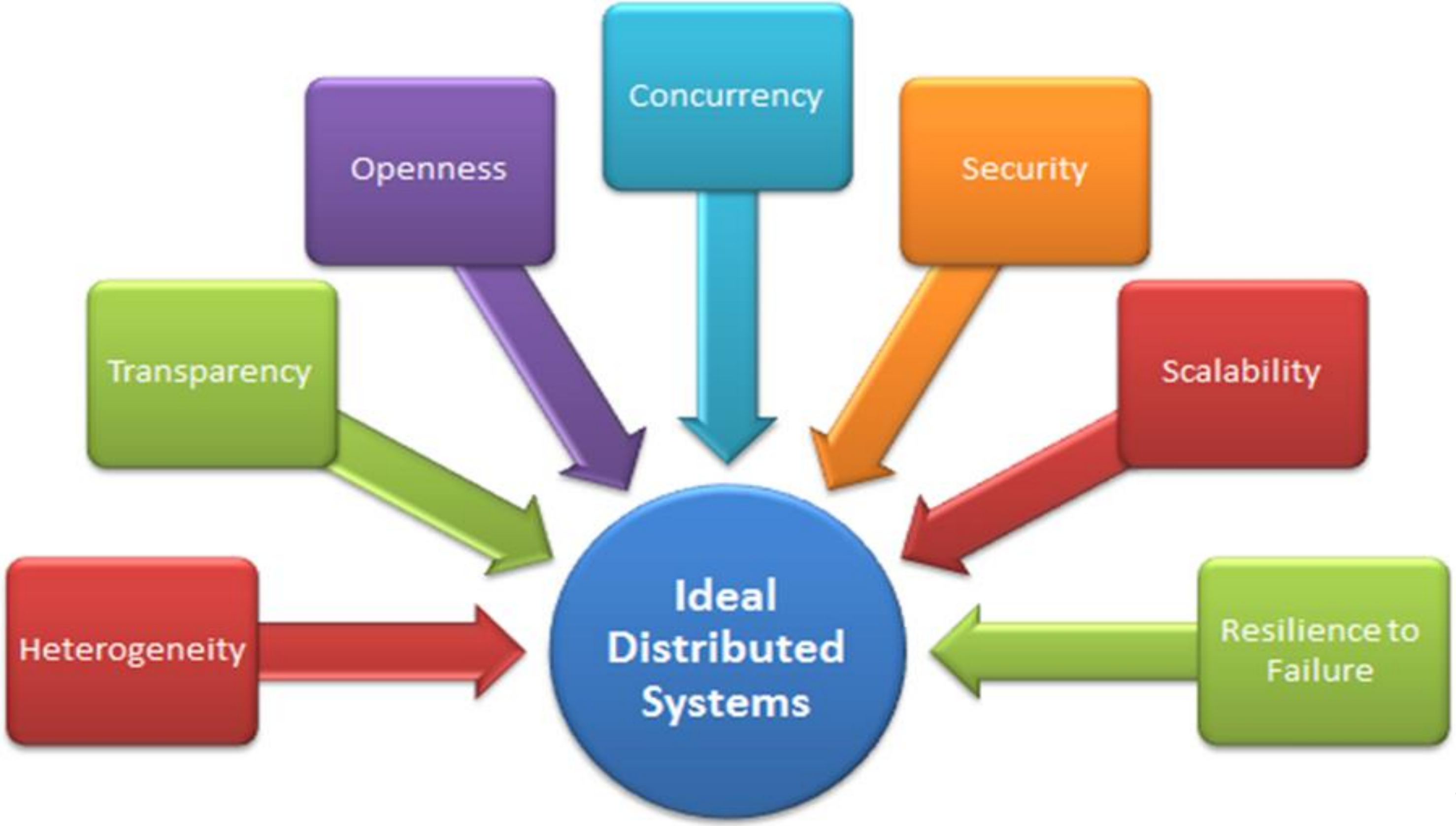
3- Security

4- Scalability

5- Failure Handling

6- Concurrency

7-Transparency



1- Heterogeneity

Heterogeneity: access all the documents are **located in different type of machines.**

- **Variety and differences :**
  - Networks
  - Computer hardware
  - Operating systems
  - Programming languages
  - Implementations by different developers

- ***Middleware*** as **software layers** to provide a programming abstraction as well as **masking the heterogeneity** of the underlying networks, hardware, OS, and programming languages

- ***Mobile Code*** to refer to code that can be sent from **one computer to another** and run at the destination (e.g., ***Java applets*** and Java *virtual machine*).

2- Openness

Openness :

how open is the software designed to be developed and shared with each other

Ex: use credit cards from different Bank on each others tellers.
their systems are open to have this flexibility

- Openness is concerned with:
  - **Extensions** and improvements of distributed systems.
  - **Detailed interfaces** of components need to be **published**
  - **Integrated** with existing components.
  - **Differences** of interface types on different processors (of different vendors) have to **be resolved**.

3- Security

Security :

- **Security is required for:**
  - Security and privacy
  - Authentication

- Clients send requests to access data managed by servers, resources in the networks:
  - Doctors requesting records from hospitals

- **New challenges**:
  - Denial of service attack
  - Security of mobile code

4- Scalability

Scalability:

how do the **computing and processing capabilities multiply** when **extended to many machines**

Ex: the Internet application , grow each day to accommodate more users like Facebook.

Adaptation of distributed systems to :

- accommodate **more users**
- **respond faster** (this is the hard one) faster processors
- **Components changed** when scale of a system increases.
- **Design components** to be scalable!

5- Failure Handling (Fault Tolerance)

Failure Handling:

how **easy and quickly to** detect **failures** in parts of the system and recovered.

Ex: Two (distributed) account databases are managed by the bank to quickly recover from a break-down.

- **Maintain *availability even at low levels of hardware/software/network***

- Fault tolerance is achieved by
  - **Recovery**
  - **Redundancy**

6- Concurrency – 7 Transparency

Concurrency:

Multiple machines can **process the same function** at **the same time**

Ex: **Multiple database users** can concurrently **access and update data in a distributed database system**. Facebook posts

- Lost updates
- Inconsistent analysis

Transparency:

- Distributed systems should be **perceived** by **users and application programmers** as a **whole** rather than as a **collection of cooperating components.**

Examples of Distributed Systems

Examples of Distributed Systems

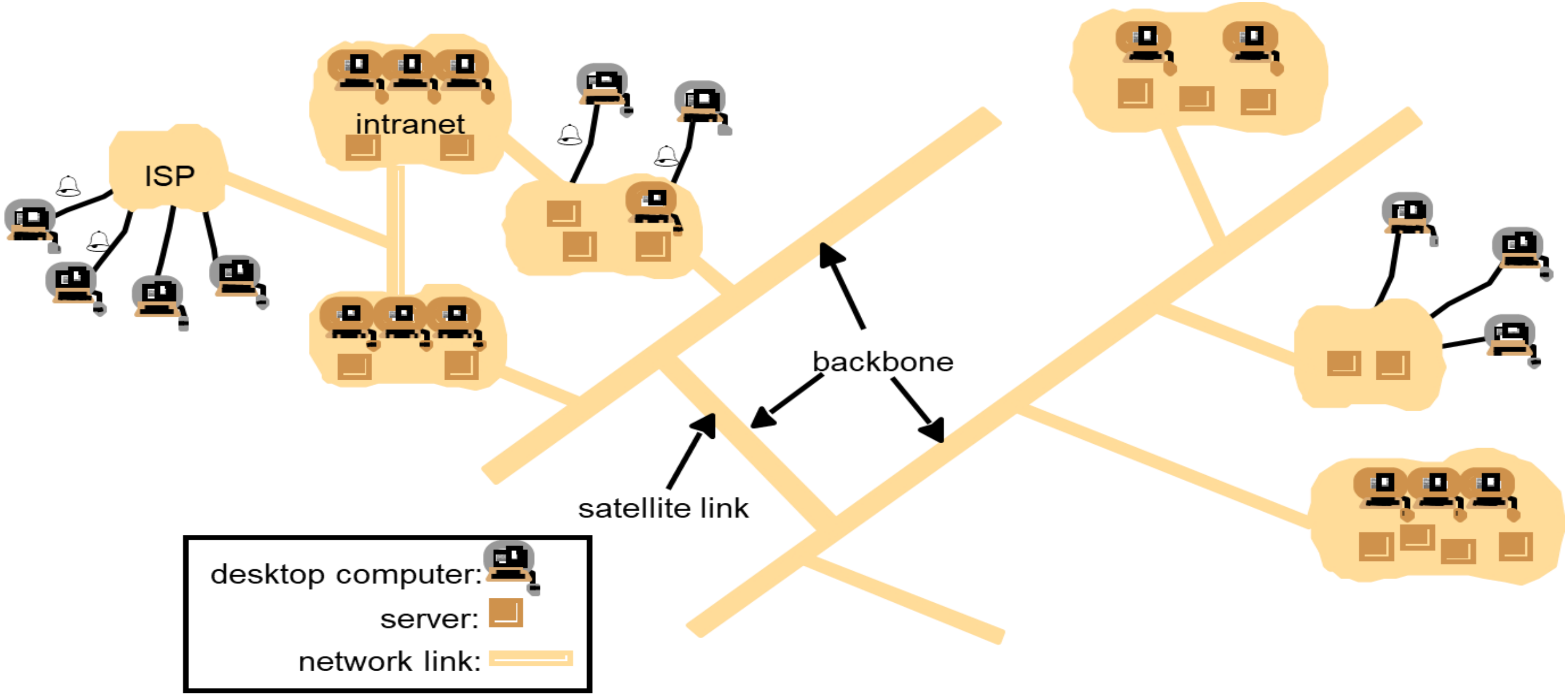
Distributed system Examples

- Domain Name System (DNS)
  - Distributed lookup table of hostname to IP address
- Facebook & Google use distributed systems extensively
  - Massive scale
  - Fast enough
  - Very reliable
- Email servers (SMTP)

Distributed system More Examples

Internet

- Internet is the largest distributed system in the world.



Session 3 (Virtualization)

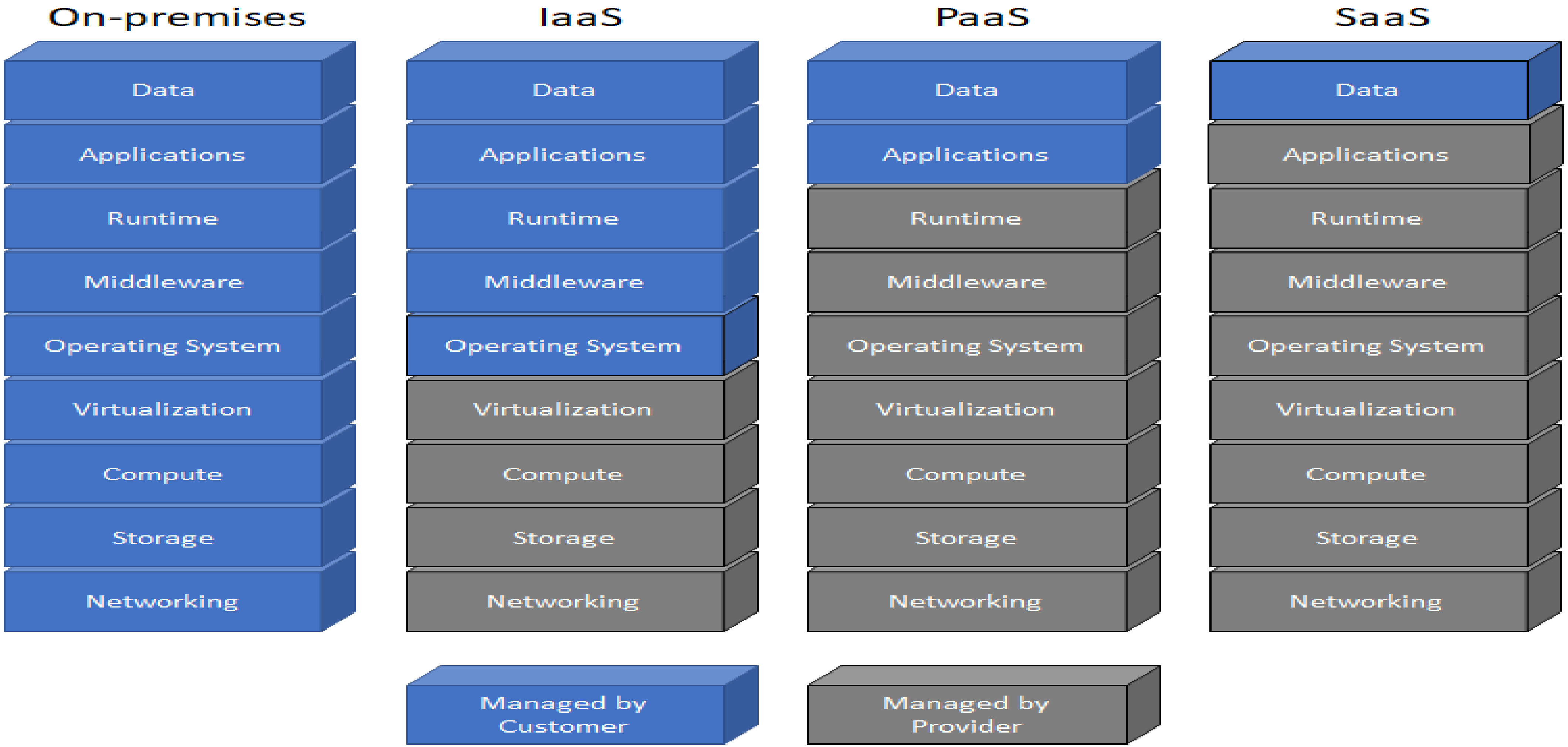
- **Virtualization**
  - Is a technology that run multiple same or different operating systems which is completely isolated from each other
  - Example: run both windows and Linux on the same machine

  - Virtualization is different from Dual Boot ?
  - Dual Boot run only one OS at the same time Virtualization run multiple OS at the same time
- **Virtualization Benefits**
  - Redundancy
  - Legacy hardware
  - Migration
  - Centralized management

Cloud computing

- **Cloud computing**
  - A pool of resources that can be rapidly provisioned in an automated, on-demand manner.
- **Value of cloud computing is :**
  - Economies of scale
  - Elastic enough to scale with the needs of your organization.
  - Cost and operational benefits
  - Easily accessed by users no matter where they reside

Cloud computing service models



On-premises	IaaS	PaaS	SaaS
Data	Data	Data	Data
Applications	Applications	Applications	Applications
Runtime	Runtime	Runtime	Runtime
Middleware	Middleware	Middleware	Middleware
Operating System	Operating System	Operating System	Operating System
Virtualization	Virtualization	Virtualization	Virtualization
Compute	Compute	Compute	Compute
Storage	Storage	Storage	Storage
Networking	Networking	Networking	Networking

Managed by Customer

Managed by Provider

Cloud computing service models

- **Software as a service (SaaS).**
  - Customers are provided access to **an application** running on a cloud infrastructure.
  - but the customer has no knowledge of, and does not manage or control, the underlying cloud infrastructure.
- **Platform as a service (PaaS).**
  - Customers can **deploy supported applications** onto the provider’s cloud infrastructure,
  - but the customer has no knowledge of, and does not manage or control, the underlying cloud infrastructure.
  - The company owns the deployed applications and data, and it is therefore responsible for the security of those applications and data.
- **Infrastructure as a service (IaaS).**
  - Customers can provision **processing, storage, networks, and other computing resources**, and **deploy and run operating systems and applications.**
  - the customer has no knowledge of, and does not manage or control, the underlying cloud infrastructure. The customer has control over operating systems, storage, and deployed applications, along with some networking components (for example, host firewalls).
  - The company owns the deployed applications and data, and it is therefore responsible for the security of those applications and data.

Cloud computing deployment models:

- **Public.**
  - A cloud infrastructure that is open to use by the general public. It’s owned, managed, and operated by a third party (or parties), and it exists on the cloud provider’s premises.
- **Community.**
  - A cloud infrastructure that is used exclusively by a specific group of organizations.
- **Private.**
  - A cloud infrastructure that is used exclusively by a single organization. It may be owned, managed, and operated by the organization or a third party (or a combination of both), and it may exist on premises or off premises.
- **Hybrid.**
  - A cloud infrastructure that comprises two or more of the aforementioned deployment models, bound by standardized or proprietary technology that enables data and application portability (for example, fail over to a secondary data center for disaster recovery or content delivery networks across multiple clouds).

Pros and Cons of Distributed Systems

- Pros
  - **Complexity is the biggest disadvantage** of distributed systems.

    There are more machines, more messages, more data being passed between more parties which leads to issues with:

    - **Data Integration & Consistency**
    - **Network and Communication Failure**
    - **Management Overhead**
- Cons
  - The ultimate **goal** of a distributed system is to **enable the scalability**, **performance and high availability** of applications.
  - **Major benefits include:**

    - **Unlimited Horizontal Scaling** - machines can be added whenever required.
    - **Low Latency** - having machines that are geographically located closer to users, it will reduce the time it takes to serve users. (google servers)
    - **Fault Tolerance** - if one server or data center goes down, others could still serve the users of the service.

Part3 lab Practices

- Use the VMware Workstation tool to host the two different
- OS on your machine

Thank You