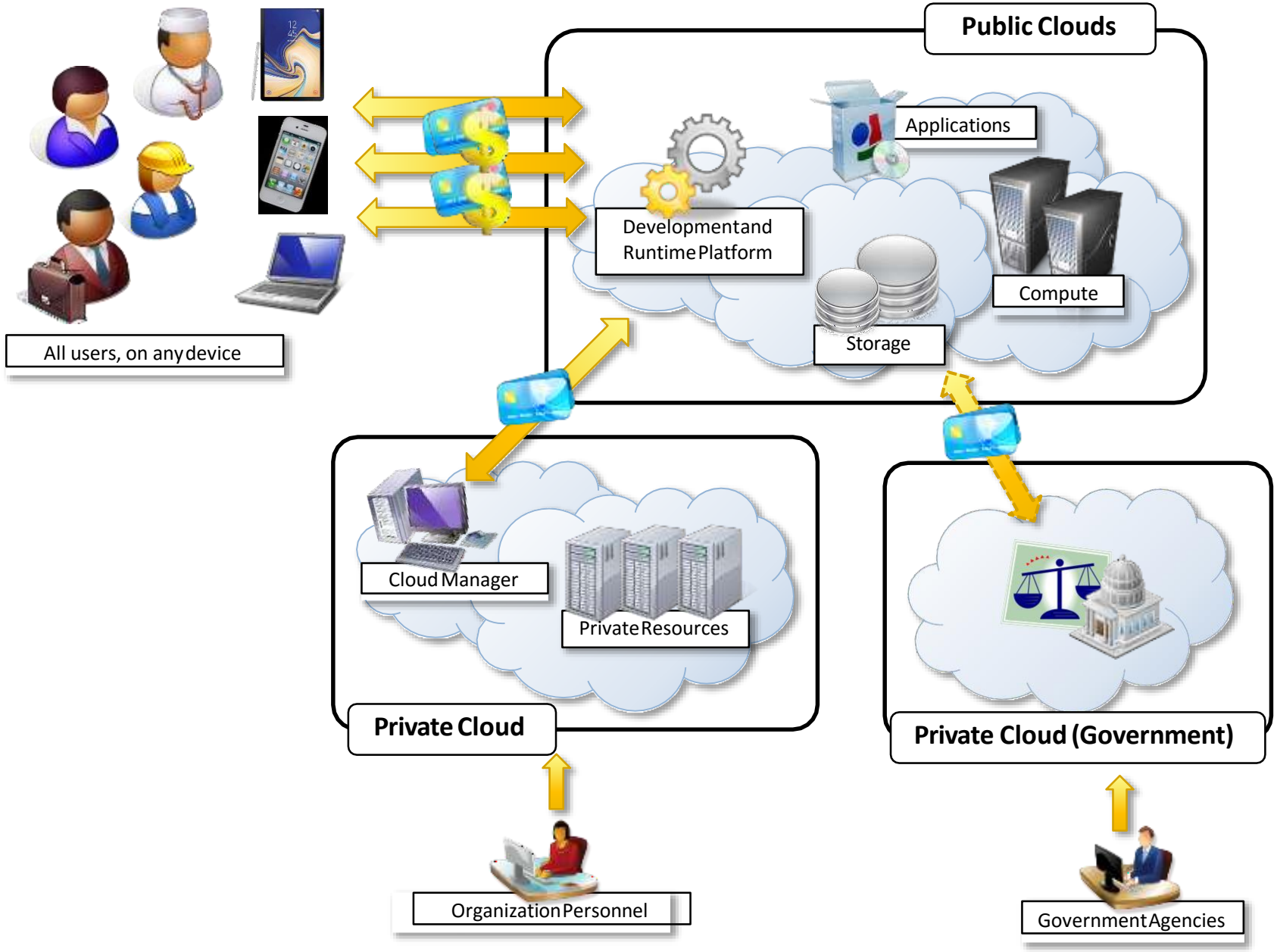

بسم الله الرحمن الرحيم

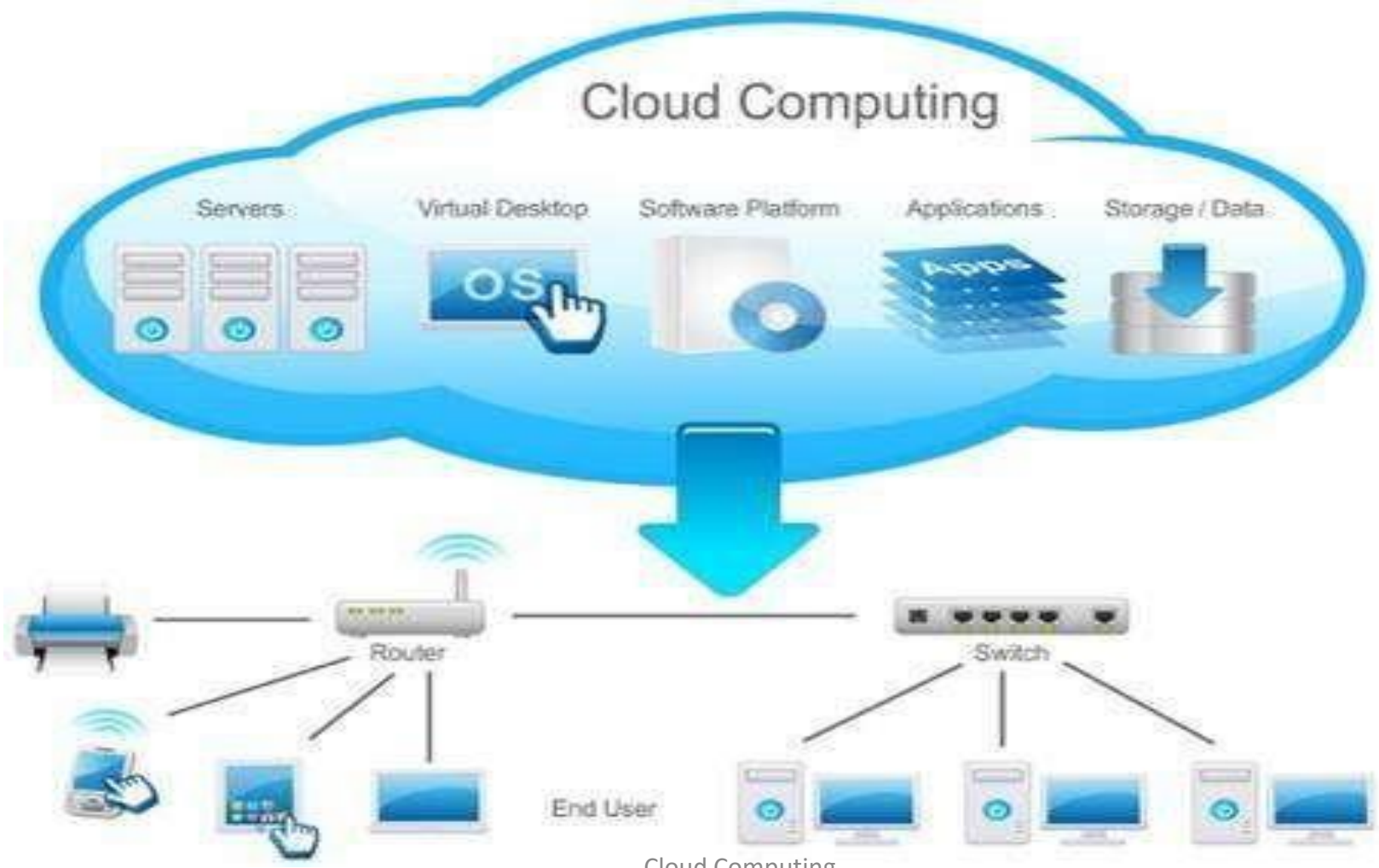
An Introduction to Cloud Computing

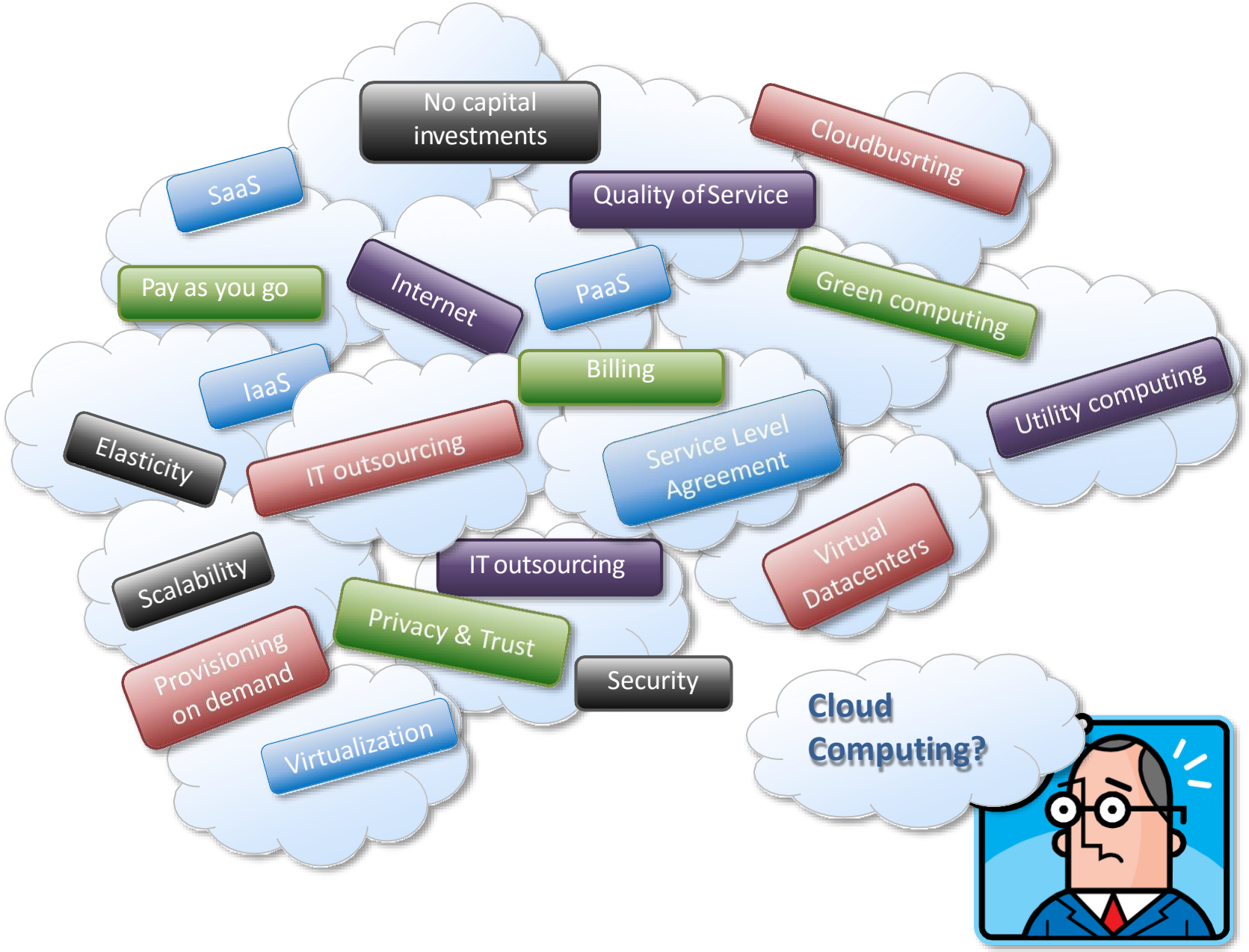
Session 1


I need to grow my infrastructure, but I do not know for how long…
I have a lot of infrastructure that I want to rent…
I have a surplus of infrastructure that I want to make use of
I cannot invest in infrastructure, I just started my business….
I have infrastructure and middleware and I can host applications
I want to focus on application logic and not maintenance and scalability issues
Global Cloud Marketplace
I have infrastructure and provide application services
I want to access and edit my documents and photos from everywhere..

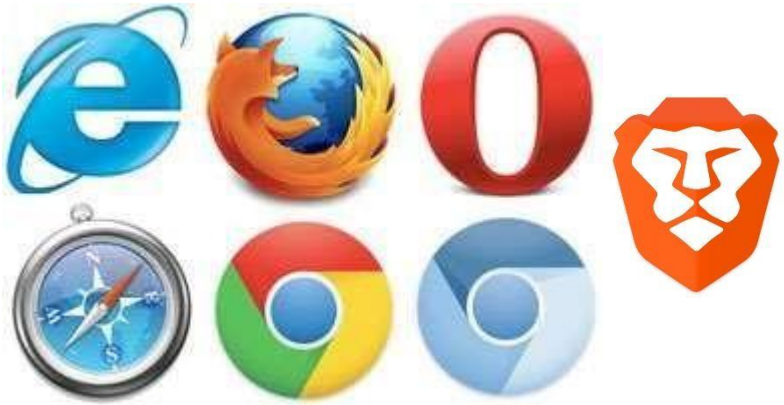

Public Clouds
Applications
Developmentand
Runtime Platform
Compute
Storage
All users, on anydevice
Cloud Manager
PrivateResources
Private Cloud
Private Cloud (Government)
OrganizationPersonnel
GovernmentAgencies

Defining Cloud




No capital investments
Cloudbusrting
SaaS
Quality of Service
Pay as you go
Internet
PaaS
Green computing
Billing
IaaS
Utility computing
Elasticity
IT outsourcing
Service Level Agreement
Virtual Datacenters
Scalability
IT outsourcing
Privacy & Trust
Provisioning on demand
Security
Virtualization
Cloud Computing?

Defining Cloud



A Closer look

Cloud computing Helping:

- Enterprises
- Governments
- Public Institutes
- Private Institutes
- Research Organization

Examples

- Large enterprise can offload some of their activities to Cloud based system.

Example

- Small Enterprises and Start-ups can afford to translate into business results their ideas more quickly without excessive upfront cost

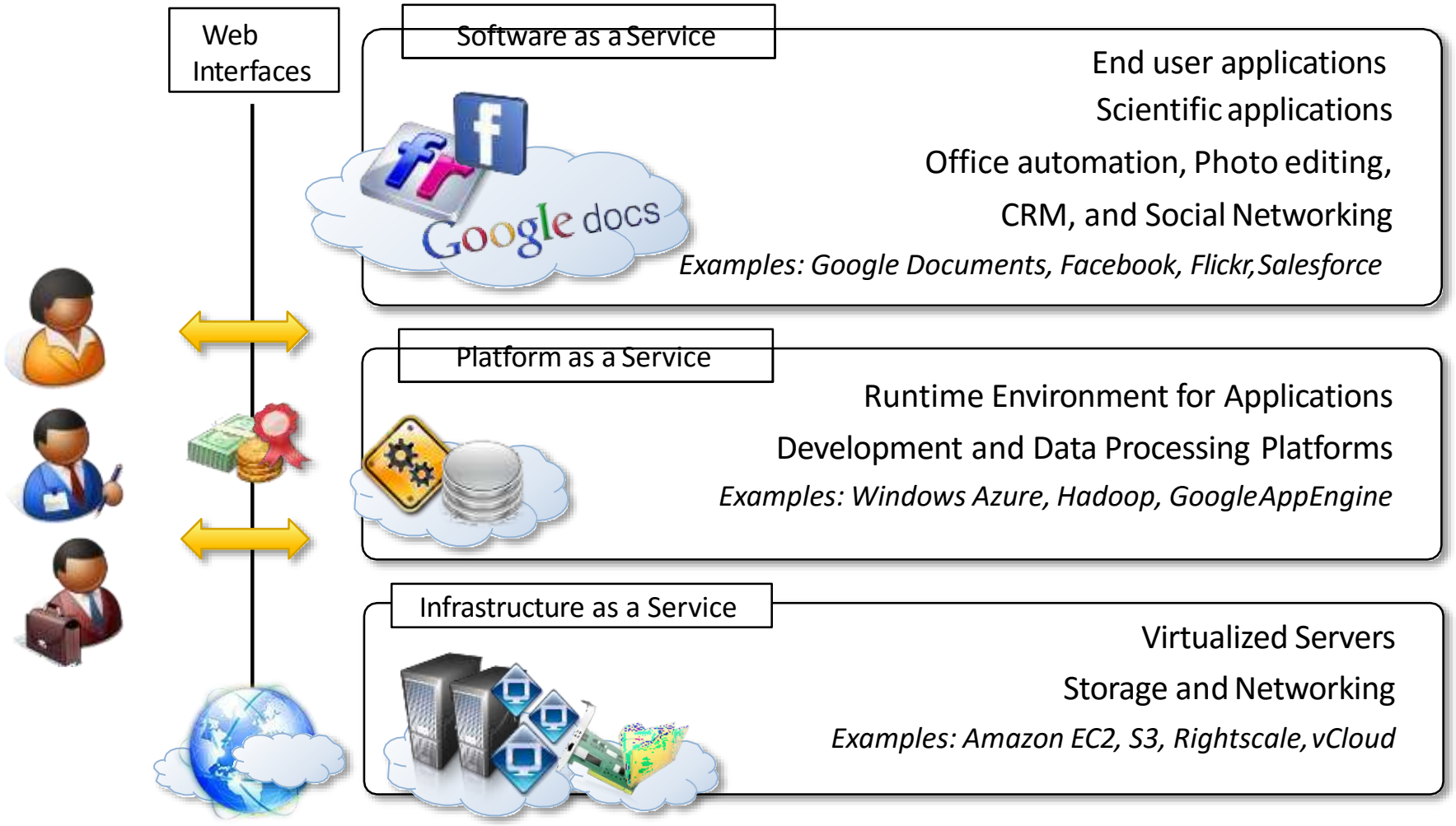
Example

- System Developers can concentrate on business logic rather than dealing with the complexity of infrastructure management and scalability

Example

- End users can have their documents accessible from everywhere and any device

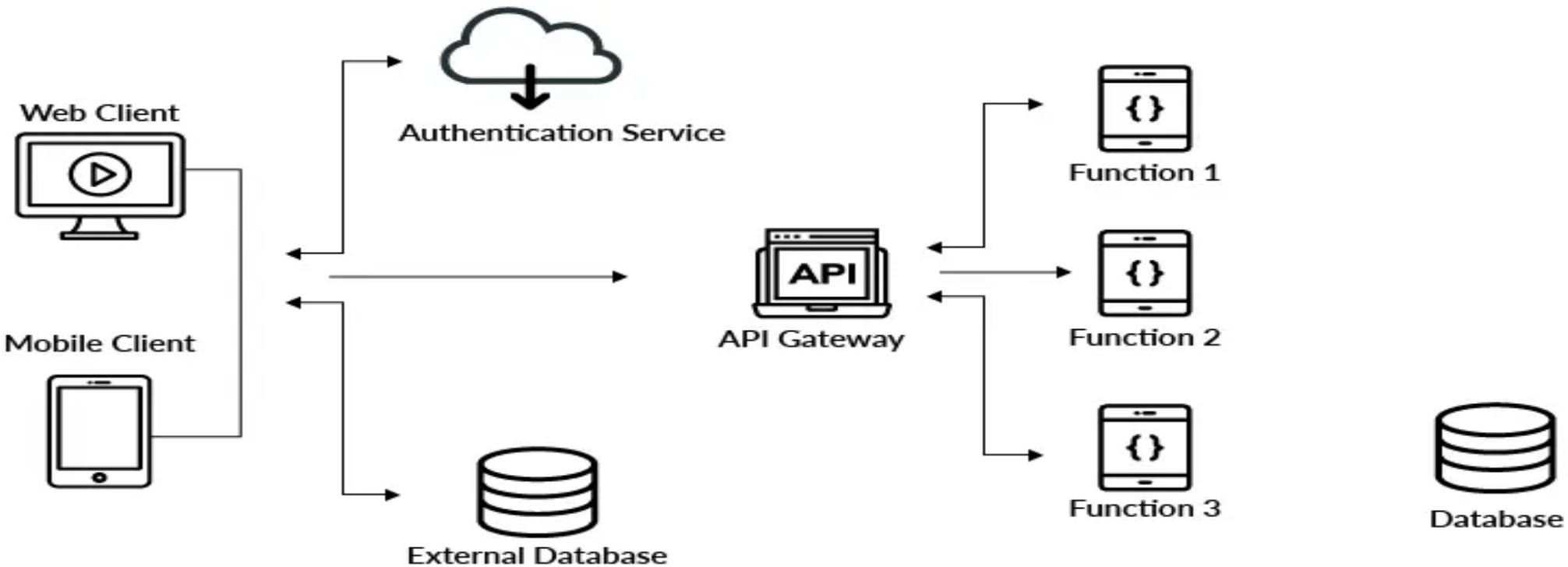
Cloud Computing Reference Model



Serverless Computing

- Serverless computing is a cloud application development and execution model that lets developers build and run code without managing servers, and without paying for idle cloud infrastructure.
- Serverless lets developers put all their focus into writing the best front-end application code and business logic they can.
- All developers need to do is write their application code and deploy it to containers managed by a cloud service provider.
- The cloud provider handles the rest, provisioning the cloud infrastructure required to run the code and scaling the infrastructure up and down on demand as needed.
- The cloud provider is also responsible for all routine infrastructure management and maintenance such as operating system updates and patches, security management, capacity planning, system monitoring and more.
- With serverless, developers never pay for idle capacity. The cloud provider spins up and provisions the required computing resources on demand when the code executes, and spins them back down again—called ‘scaling to zero’—when execution stops.
- The billing starts when execution starts, and ends when execution stops; typically, pricing is based on execution time and resources required.

Working Of Serverless Architecture



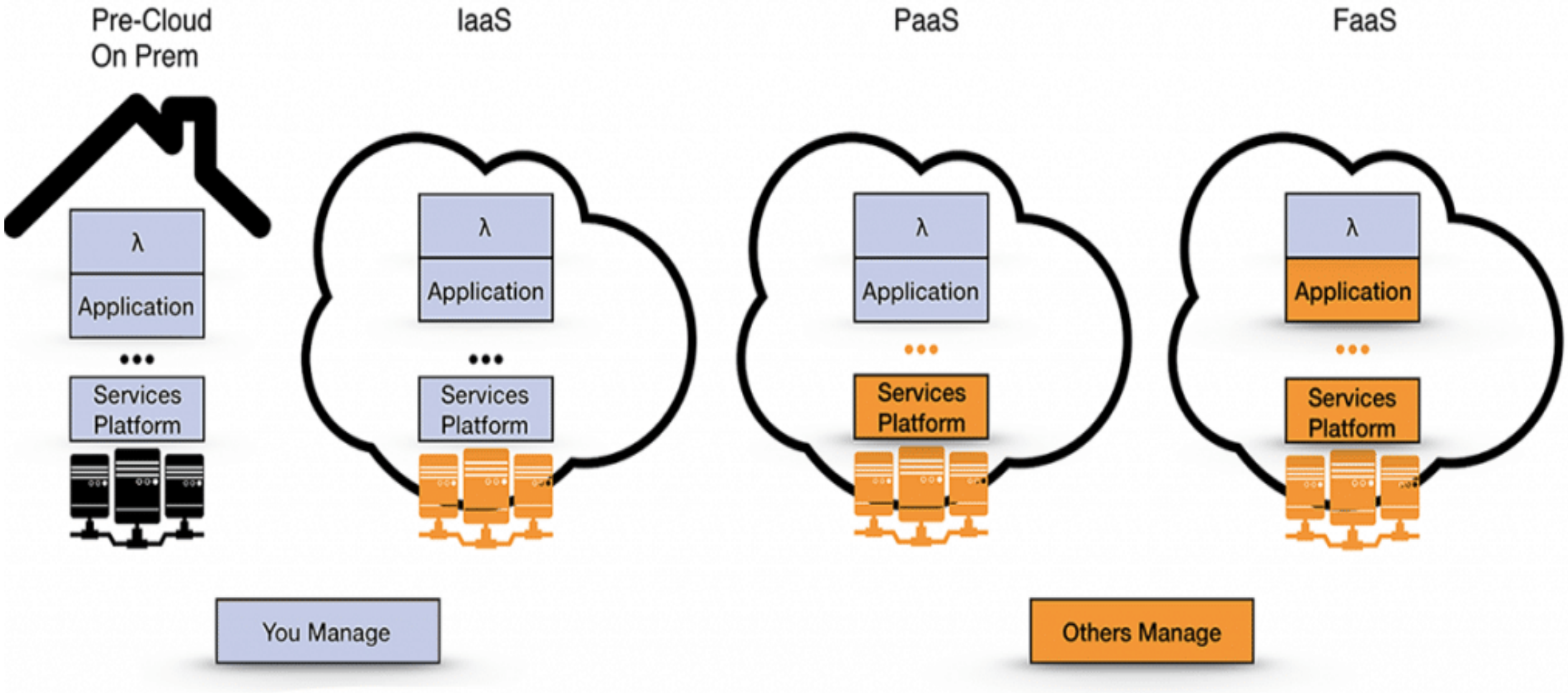
FaaS

- FaaS is a subset of serverless.
- FaaS, or Function-as-a-Service, is a cloud-computing service that allows customers to execute code in response to events, without managing the complex infrastructure typically associated with building and launching microservices applications.
- Hosting a software application on the internet typically requires provisioning and managing a virtual or physical server and managing an operating system and web server hosting processes.
- With FaaS, the physical hardware, virtual machine operating system, and web server software management are all handled automatically by the cloud service provider. This allows developers to focus solely on individual functions in their application code.

Advantages of FAAS

- Pay for execution
- Auto Scalable
- Faster Time to Market
- Polyglot Environment (Support for multiple programming languages)
- Highly Available

Evolution of Functions as a Service



FaaS vs. serverless

- Serverless and Functions-as-a-Service (FaaS) are often conflated with one another but the truth is that FaaS is actually a subset of serverless.
- Serverless is focused on any service category, be it compute, storage, database, messaging, api gateways, etc. where configuration, management, and billing of servers are invisible to the end user.
- FaaS, on the other hand, while perhaps the most central technology in serverless architectures, is focused on the event-driven computing paradigm wherein application code, or containers, only run in response to events or requests.

FaaS/Serverless computing providers

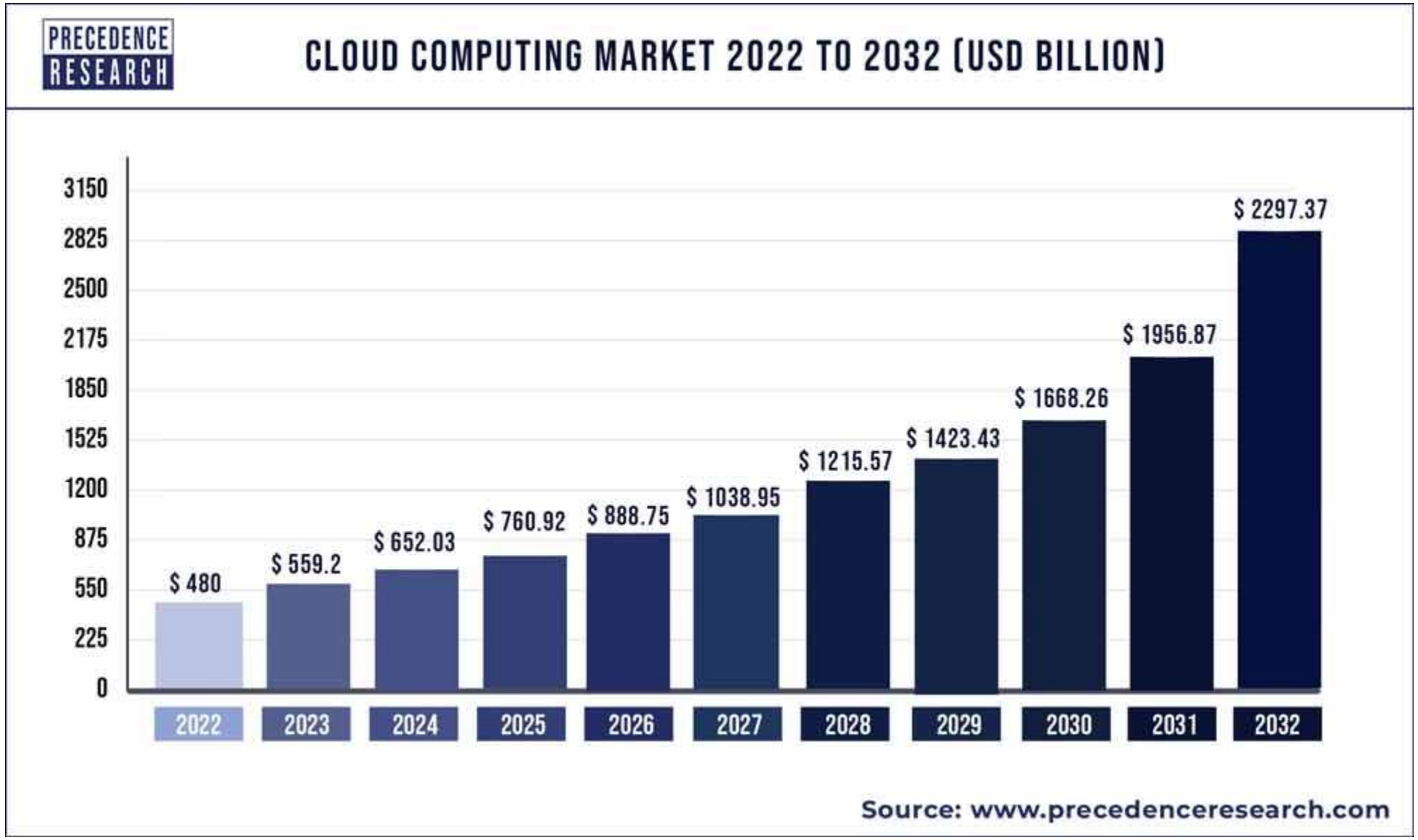
1. **Amazon Web Services (AWS) Lambda** is a popular serverless computing platform. It allows developers to write functions in various programming languages, including **Node.js, Python, Java, and C#**. These functions can be triggered by events such as HTTP requests, database updates, file uploads, or scheduled events.
2. **Azure Functions** is Microsoft's serverless computing offering. It supports languages like **C#, JavaScript, PowerShell, Python, and TypeScript**. Developers can write functions that respond to various triggers, including HTTP requests, timers, and messages from Azure services like Event Grid or Service Bus. Azure Functions automatically scales the resources to handle the workload, and users are billed based on the number of executions and the execution time.
3. **Google Cloud Functions** is a serverless computing platform provided by Google Cloud Platform (GCP). It supports languages such as **Node.js, Python, and Go.** Functions can be triggered by events from various sources like HTTP requests, Pub/Sub messages, or changes in Cloud Storage or Firestore. Google Cloud Functions scales automatically and charges users based on the number of function invocations, execution time, and memory consumption.
4. **IBM Cloud Functions** is the serverless computing offering from IBM Cloud. It supports multiple programming languages, including **Node.js, Swift, Python, and Java**. Users can create functions that respond to events from various sources like HTTP requests, database changes, or message queues. IBM Cloud Functions provides autoscaling capabilities and pricing is based on the number of function invocations and the execution time.

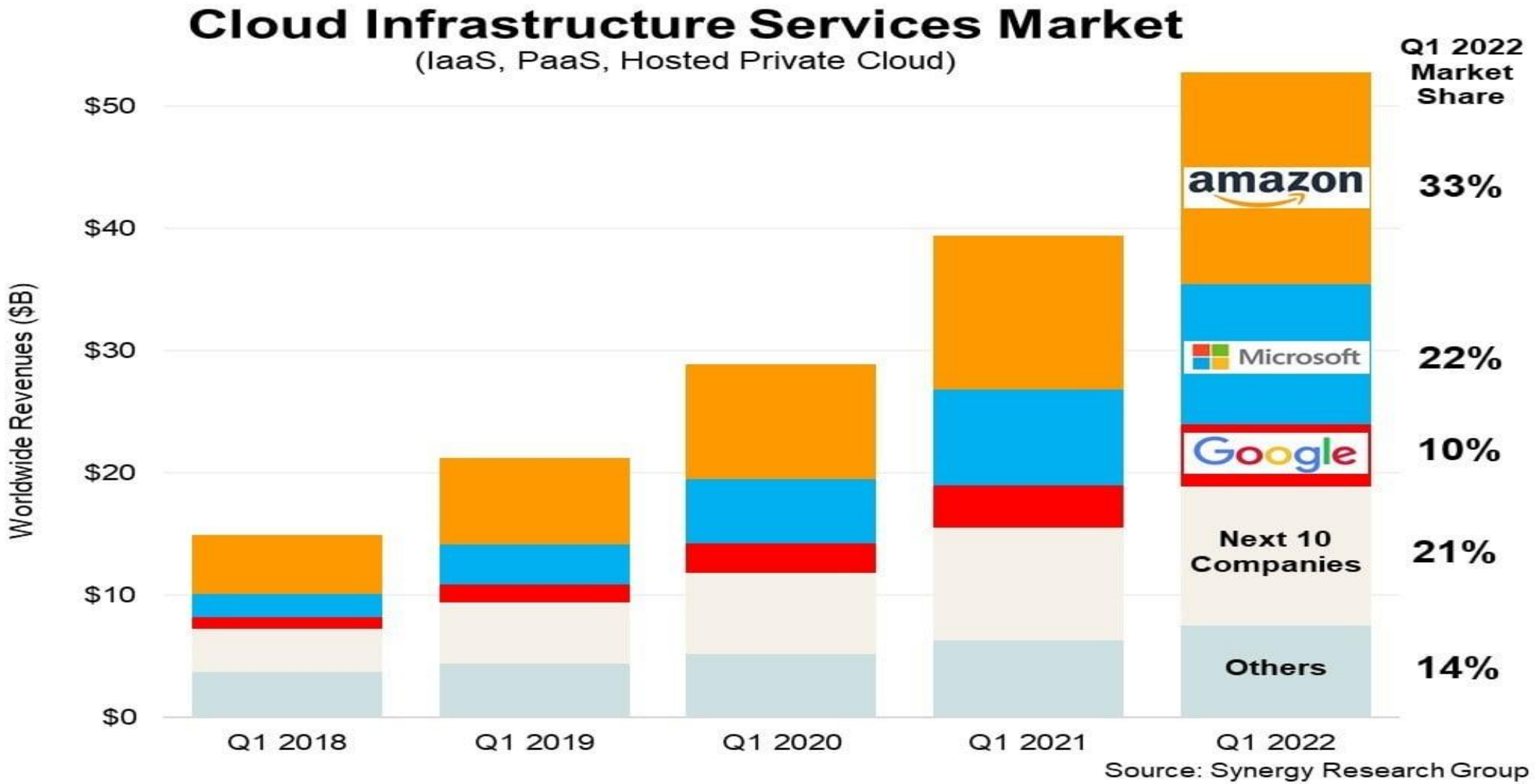
Cloud Computing Characteristics and Benefits

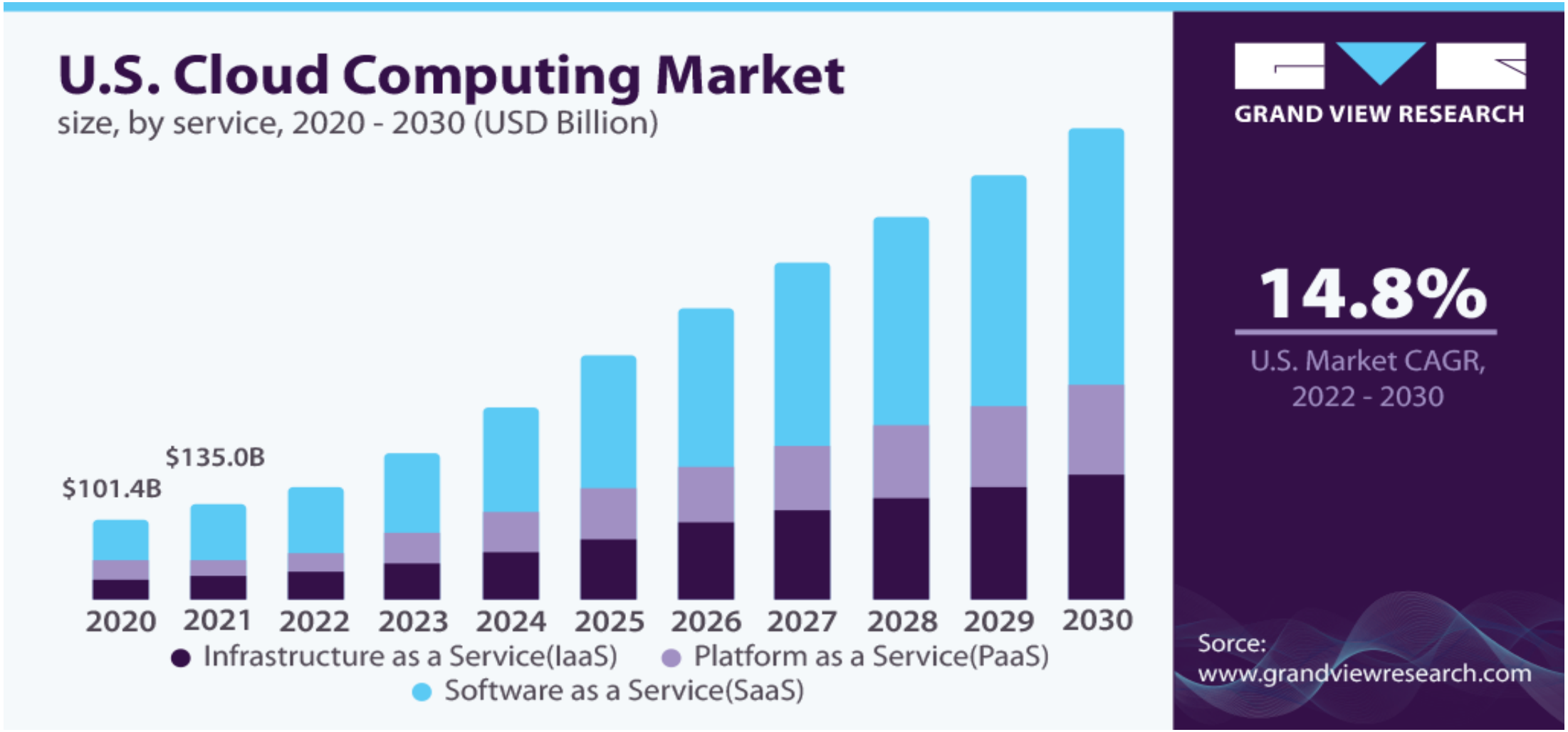
- No Upfront Commitments
- On demand access
- Nice pricing
- Simplified application acceleration and scalability
- Efficient resource allocation
- Energy efficiency and seamless creation and use third-party services.

Challenges Ahead

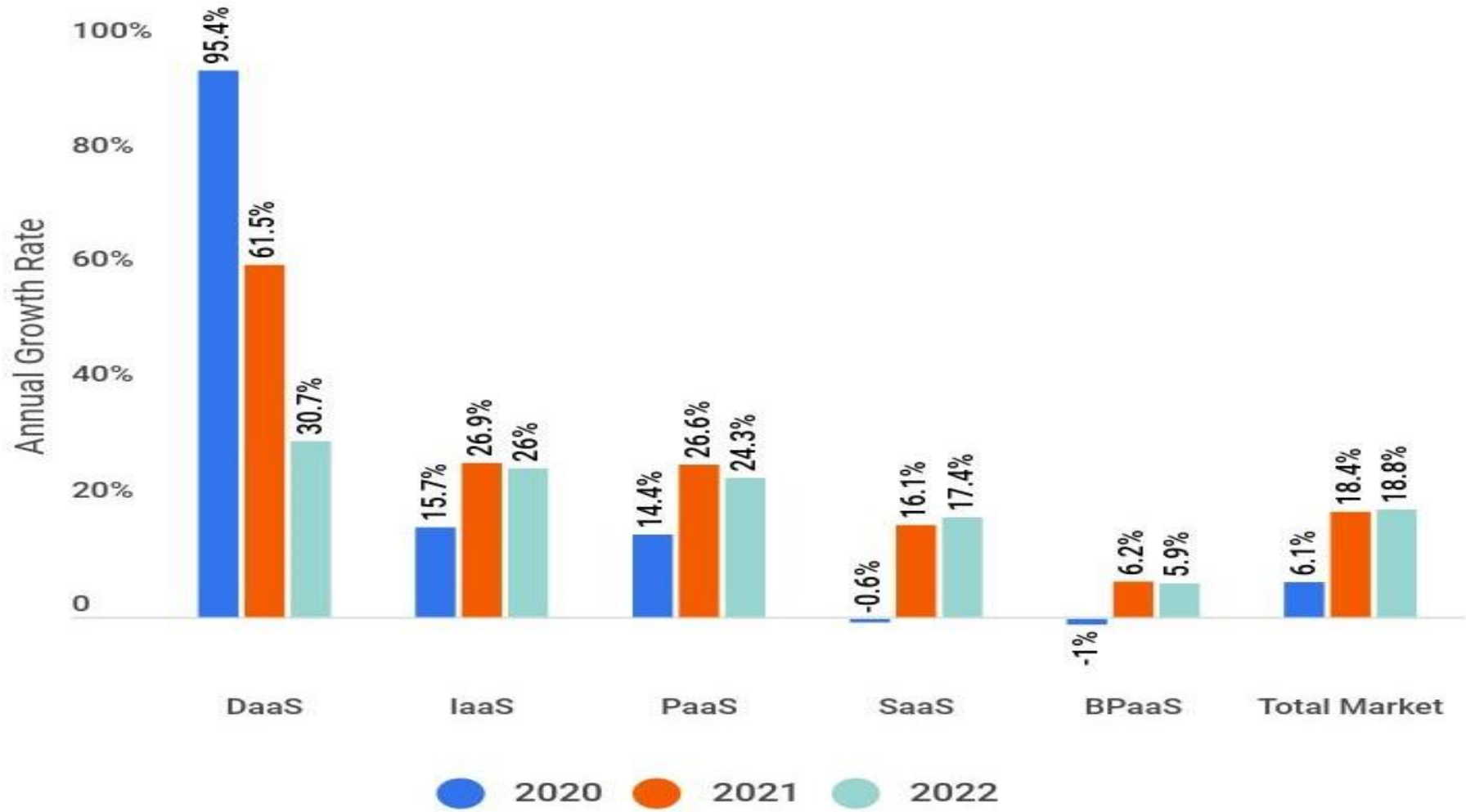
- Dynamic Provisioning of Cloud Computing Services
- Security and Privacy
- Legal issues
- Performance and Bandwidth Cost
- Reliability and Availability

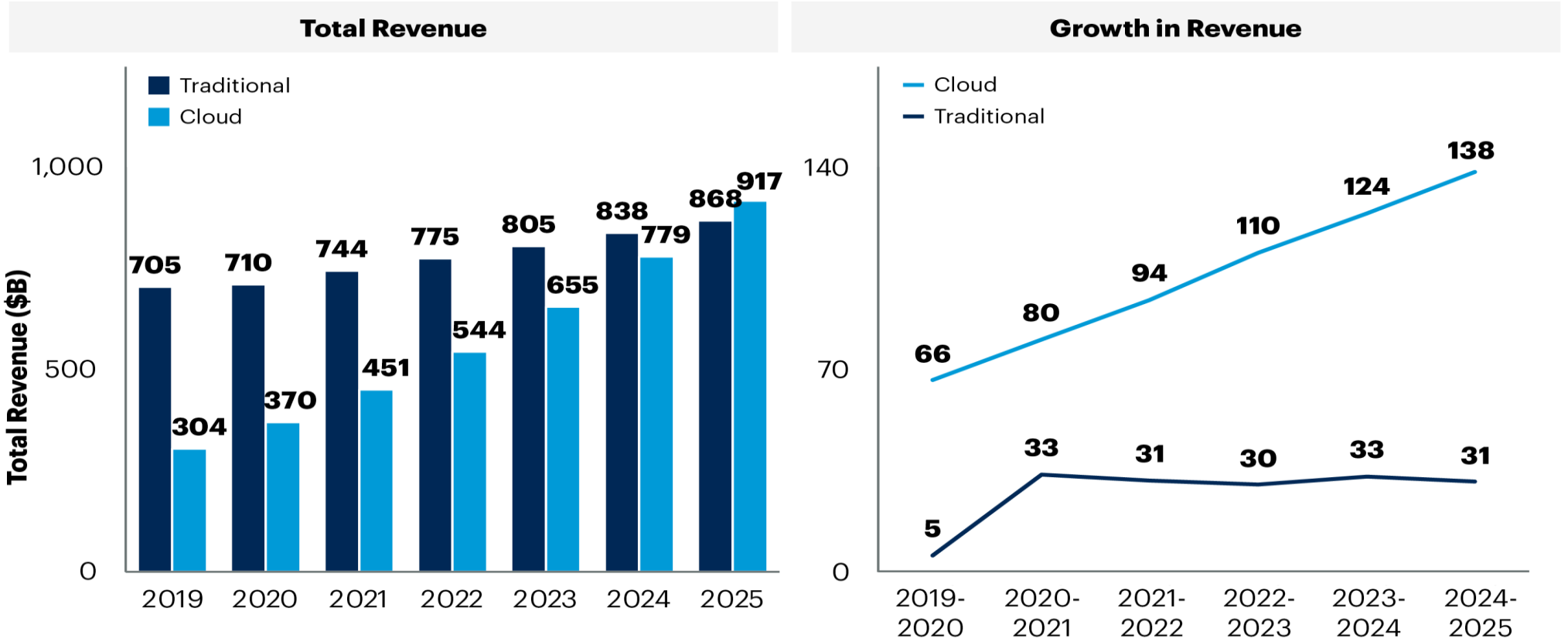

PRECEDENCE RESEARCH
CLOUD COMPUTING MARKET 2022 TO 2032 (USD BILLION)
3150
2825
2500
2175
1850
1525
1200
875
550
225
0
$ 480
$ 559.2
$ 652.03
$ 760.92
$ 888.75
$ 1038.95
$ 1215.57
$ 1423.43
$ 1668.26
$ 1956.87
$ 2297.37
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
Source: www.precedenceresearch.com


Cloud Infrastructure Services Market
(IaaS, PaaS, Hosted Private Cloud)
Worldwide Revenues ($B)
$50
$40
$30
$20
$10
$0
Q1 2018
Q1 2019
Q1 2020
Q1 2021
Q1 2022
Q1 2022 Market Share
amazon
33%
Microsoft
22%
Google
10%
Next 10 Companies
21%
Others
14%
Source: Synergy Research Group


U.S. Cloud Computing Market
size, by service, 2020 - 2030 (USD Billion)
$101.4B
$135.0B
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
Infrastructure as a Service(IaaS)
Platform as a Service(PaaS)
Software as a Service(SaaS)
GRAND VIEW RESEARCH
14.8%
U.S. Market CAGR, 2022 - 2030
Sorce: www.grandviewresearch.com

PUBLIC CLOUD SERVICES ANNUAL GROWTH RATES

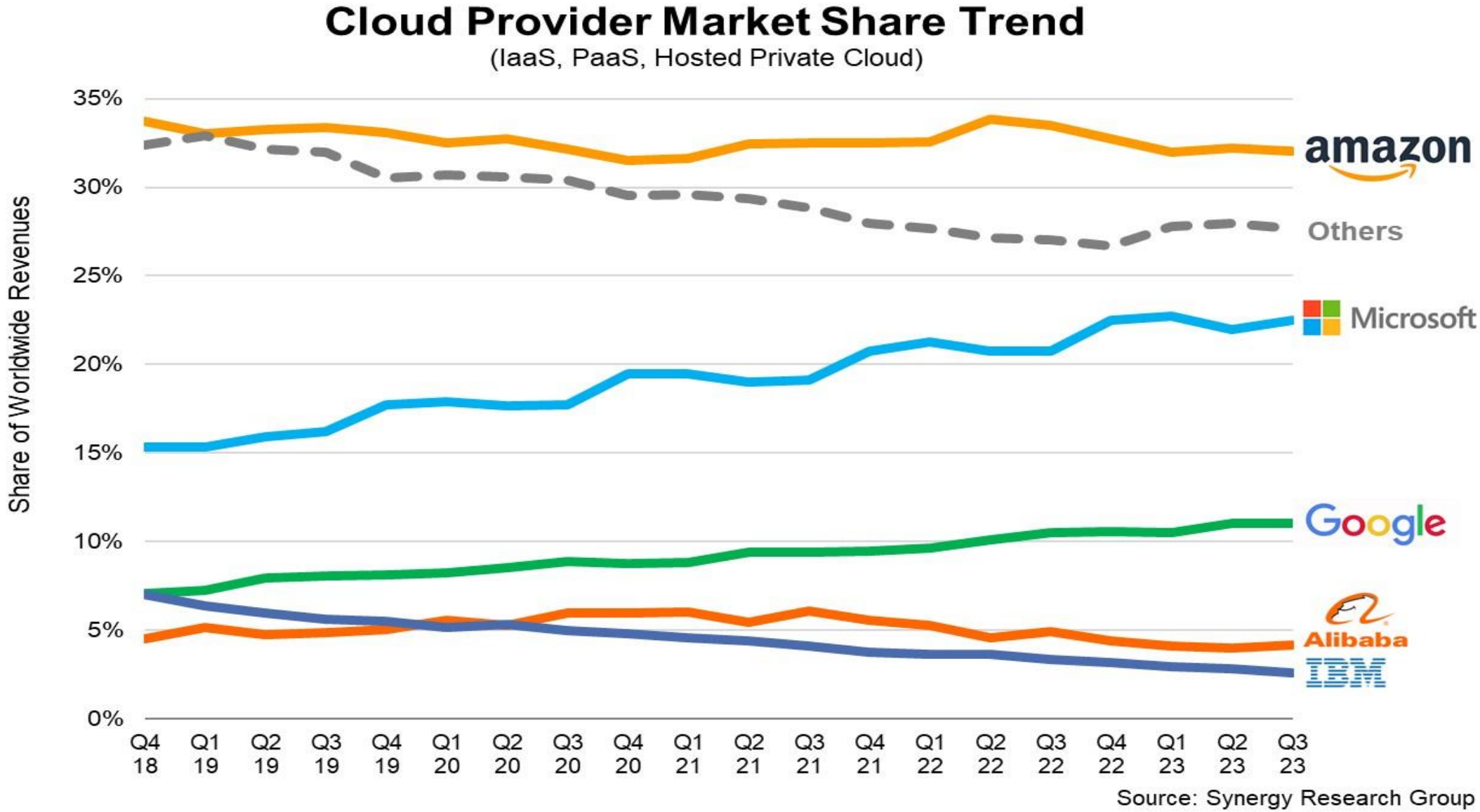


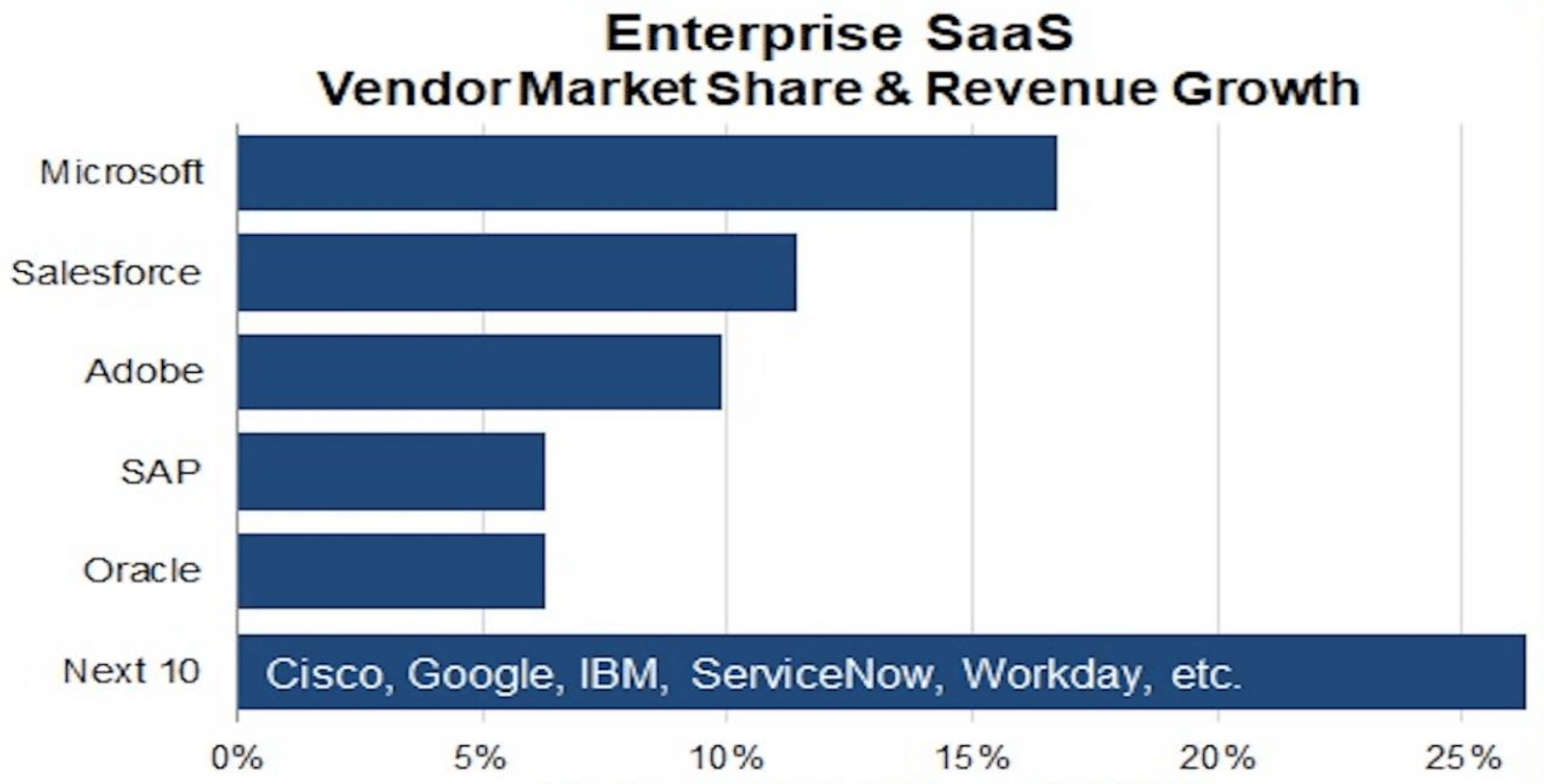


Source: Gartner

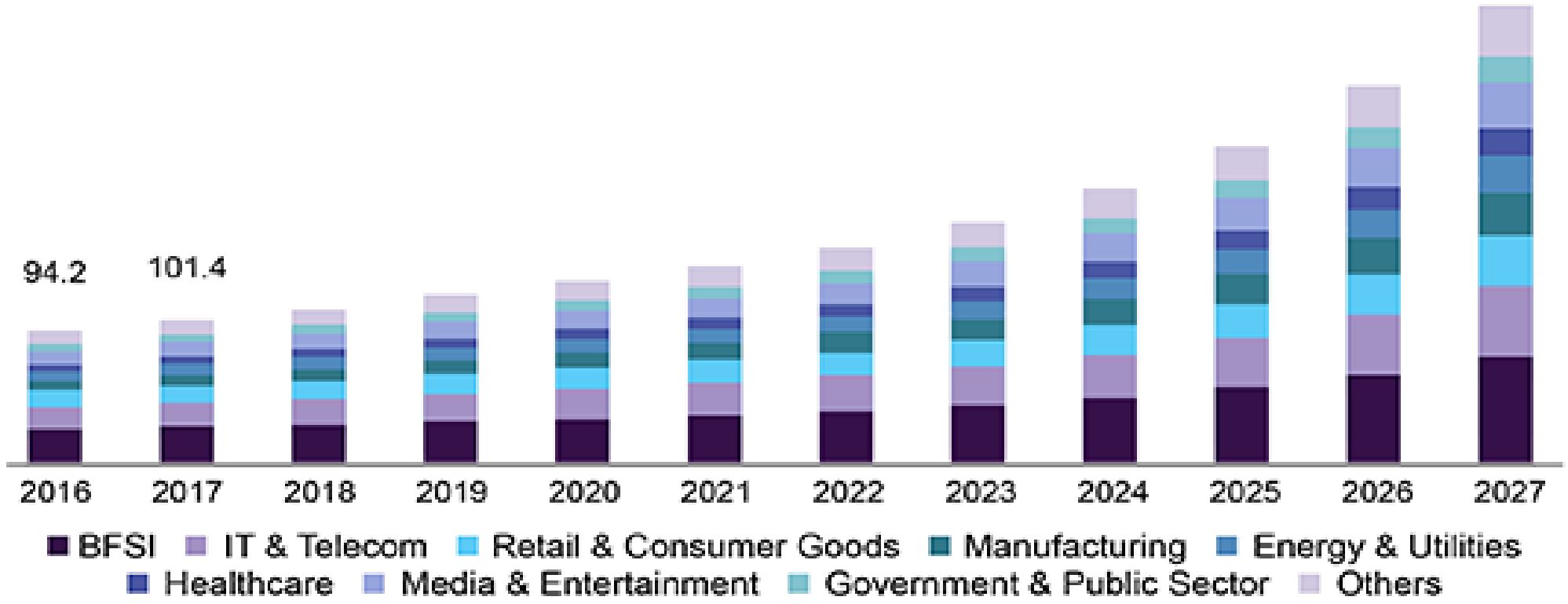
758067_C

Gartner


Cloud Provider Market Share Trend
(IaaS, PaaS, Hosted Private Cloud)
Share of Worldwide Revenues
35%
30%
25%
20%
15%
10%
5%
0%
Q4 18
Q1 19
Q2 19
Q3 19
Q4 19
Q1 20
Q2 20
Q3 20
Q4 20
Q1 21
Q2 21
Q3 21
Q4 21
Q1 22
Q2 22
Q3 22
Q4 22
Q1 23
Q2 23
Q3 23
amazon
Others
Microsoft
Google
Alibaba
IBM
Source: Synergy Research Group


Enterprise SaaS
Vendor Market Share & Revenue Growth
Microsoft
Salesforce
Adobe
SAP
Oracle
Next 10
Cisco, Google, IBM, ServiceNow, Workday, etc.
0%
5%
10%
15%
20%
25%

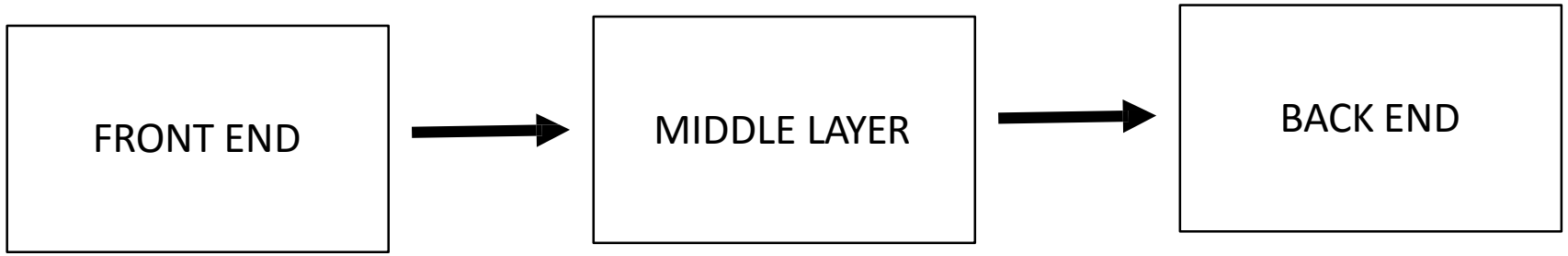
U.S. cloud computing market size, by end use, 2016 - 2027 (USD Billion)



Source: www.grandviewresearch.com

HOW CLOUD WORKS

- THREE TIER CLOUD SYSTEM



FRONT END:

- Is the part of the system that we see everyday.
- This interface could be in the form of applications installed on our devices or websites we retrieve through our web browsers.

HOW CLOUD WORKS

MIDDLE LAYER:

- Isn’t necessarily physical. It is represented by software on both sides of the front end and the back end, and we call this middleware.
- Middleware is software that is used on both ends to ensure the front and the back can speak to each other.
- Administered by a central server and follows a specific sets of rules, known as protocols.

BACK END:

- It’s where the actual cloud part of the system is housed.
- This is made of computers, servers, and databases and storage systems.

Types of scaling

The following are the types of scaling:

1. Horizontal Scaling-scaling out and scaling in.

   The allocating or releasing of IT resources that are of the same type is referred to as horizontal scaling. The horizontal allocation of resources is referred to as scaling out and the horizontal releasing of resources is referred to as scaling in.

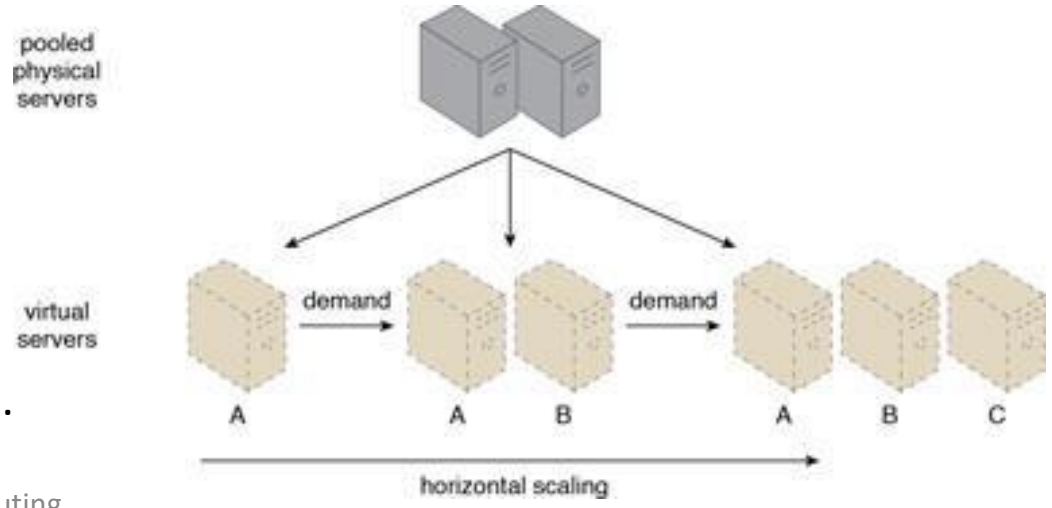


Figure 1 - An IT resource (Virtual Server A) is scaled out by adding more of the same IT resources (Virtual Servers B and C).

Types of scaling

2. Vertical Scaling- scaling up and scaling down

When an existing IT resource is replaced by another with higher or lower capacity, vertical scaling is considered o have occurred.

To replace an IT resource with another that has a higher is referred to as scaling up and the replacing an IT resource with another that has a lower capacity is considered scaling down.

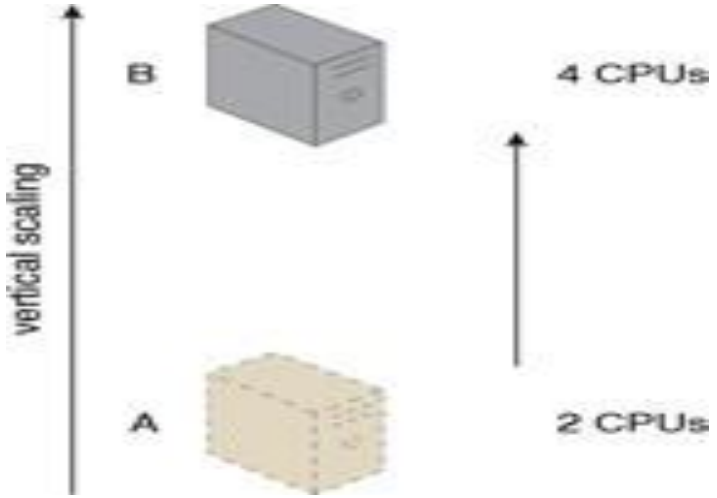


Figure 2 - An IT resource (a virtual server with two CPUs) is scaled up by replacing it with a more powerful IT resource with increased capacity for data storage (a physical server with four CPUs).

Horizontal vs Vertical Scaling

Horizontal Scaling	Vertical Scaling
less expensive (through commodity hardware components)	more expensive (specialized servers)
IT resources instantly available	IT resources normally instantly available
resource replication and automated scaling	additional setup is normally needed
additional IT resources needed	no additional IT resources needed
not limited by hardware capacity	limited by maximum hardware capacity

Table 1 - A comparison of horizontal and vertical scaling.

Evolution of cloud technologies

- *Distributed Systems*

→A distributed system is a collection of independent computers that appears to its users as a single system and also it acts as a single computer.

→The main and primary motive of distributed systems is to share resources and to utilize them better.

→This is absolutely true in case of cloud computing because in cloud computing we are sharing the single resource by paying rent.

→The resource is single because the definition of cloud computing clearly states that in cloud computing the single central copy of a particular software is stored in a sever (which is located on a anonymous location) and users are accessing that on PAY PER USE BASIS.

Characteristics of Distributed Applications

- Modern applications consume data from distributed data sources to fulfill users expectations
- Distributed applications are designed for:
  - Scalability
  - Low latency
  - Availability
  - Reliability
  - Security and privacy

Logical Layers of Distributed Applications

- Separation of concerns (Separating the responsibilities) between different components helps to achieve better maintainability, testability, and agility.
- It is easier to test each layer separately rather than testing the whole system together.
- Applications are designed in layers:
  1. Data Layer
  2. Business Layer
  3. User Interface Layer
  4. Service Layer

Data Layer

- The data layer is responsible for storing, querying, updating, or deleting the data as required while maintaining a reasonable performance.
- This can be a complicated task when you are dealing with a large set of data, distributed across several data sources.
- Data can be replicated, distributed, and handled according to its characteristics. For example, client contacts can be replicated across the data center because they change slowly. However, information about stocks must be always accurate and therefore must be read from a single source.

Business Layer

- The business layer(execution layer) contains the business logic and is responsible for carrying out the use-case scenarios of the application.
- This layer implements the logic of the application.
- The business logic uses the data layer to read and store data, and the UI layer to interact with the client.
- The execution layer contains all the algorithms and logic of the application and is considered the brain of the application.

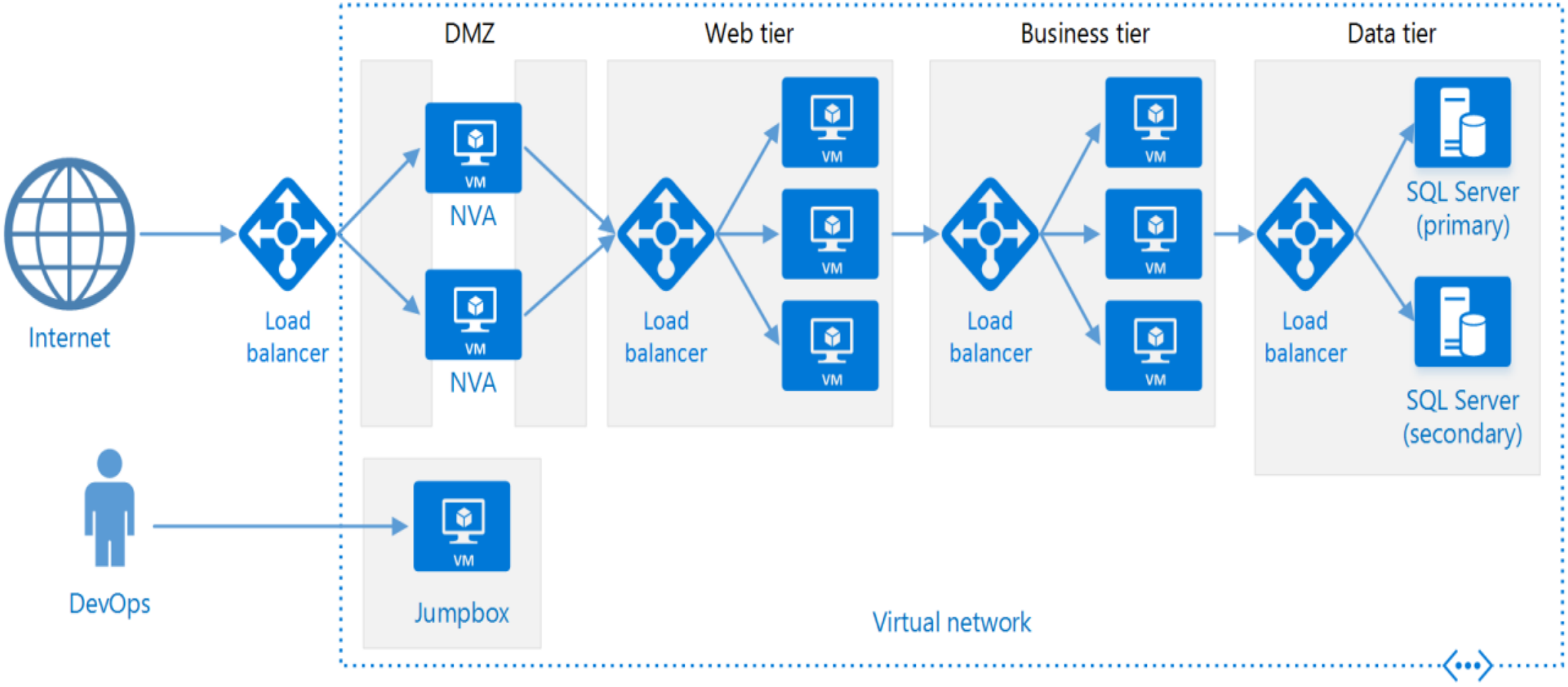
Service Layer

- The service layer exposes some of the capabilities of the application to the world as services.
- Other applications might consume these services and use them as a data source or as a remote execution engine.
- The service layer acts as the interface for other applications, in contrast to the user interface layer, which targets humans.
- The service layer drives collaboration of applications and enables distribution of computing load and data.
- It is responsible for defining a contract that consumers must maintain to use the service.
- It enforces security policies, validates incoming requests, and maintains the application resources.

User Interface Layer

- The user interface layer is the layer through which users interact with the application.
- It visually depicts the data and operations of the application effectively and provides the users a simplified medium for consuming the application data.
- While designing the UI, developers must consider the varying expectations of different people and cultures.
- The UI should always be responsive, yet its ability to respond quickly might be CPU-intensive especially when using modern interfaces such as touchscreens.
- The UI can be displayed on a variety of different devices, some of which might have extreme limitations such as screen size and resolution.
- Nevertheless, the UI must be effective and present a useful visualization.
- The UI must provide simple, yet effective methods for the user to enter data and activate the business and data layers to store and process it. Proper UI design is crucial because if the UI is not user friendly, the application will not be used.

Three tiers Applications in Cloud



Mainframes

- A large high-speed computer, especially one supporting numerous workstations or peripherals.
- The central processing unit and primary memory of a computer.

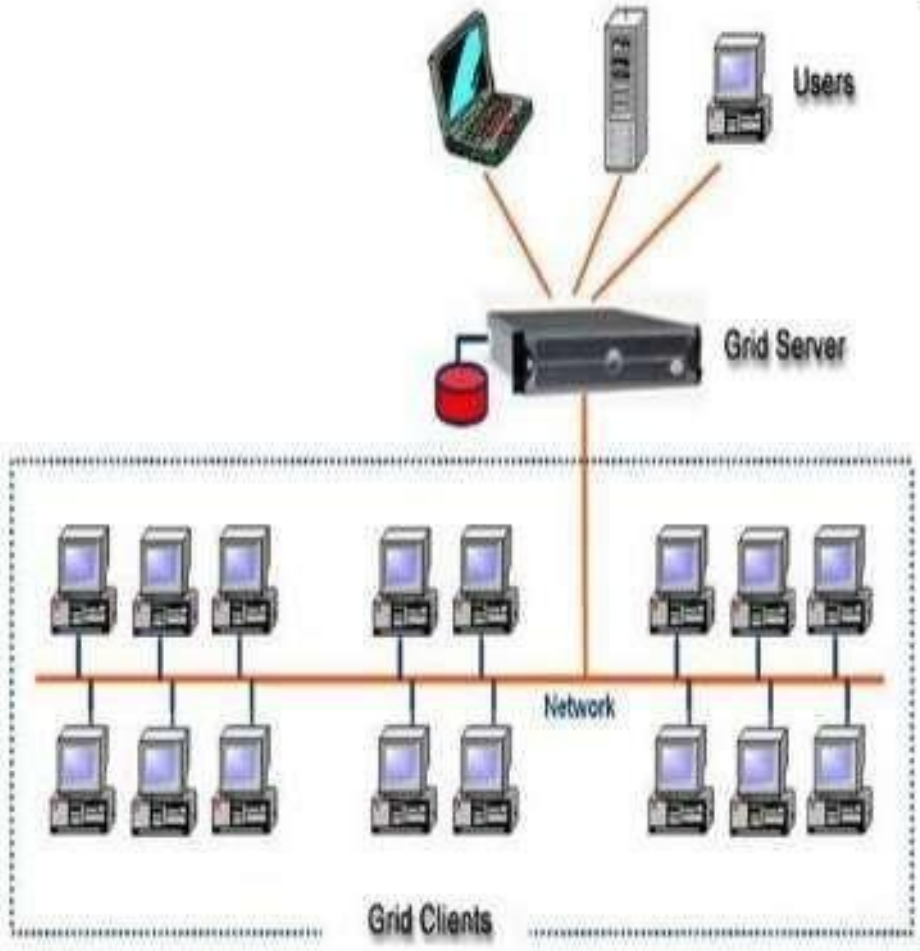
Clusters

- A **computer cluster** consists of a set of loosely or tightly connected computers that work together so that, in many respects, they can be viewed as a single system.
- computer clusters have each node set to perform the same task, controlled and scheduled by software.

Grids

- **Grid computing** is the collection of **computer** resources from multiple locations to reach a common goal. The **grid** can be thought of as a **distributed** system with non- interactive workloads that involve a large number of files.
- The components of a cluster are usually connected to each other through fast local area networks, with each *node* (computer used as a server) running its own instance of an operating system. In most circumstances,

How Grid computing works ?



In general, a grid computing system requires:

- **At least one computer, usually a server, which handles all the administrative duties for the System**
- **A network of computers running special grid computing network software.**
- **A collection of computer software called middleware**

Virtualization

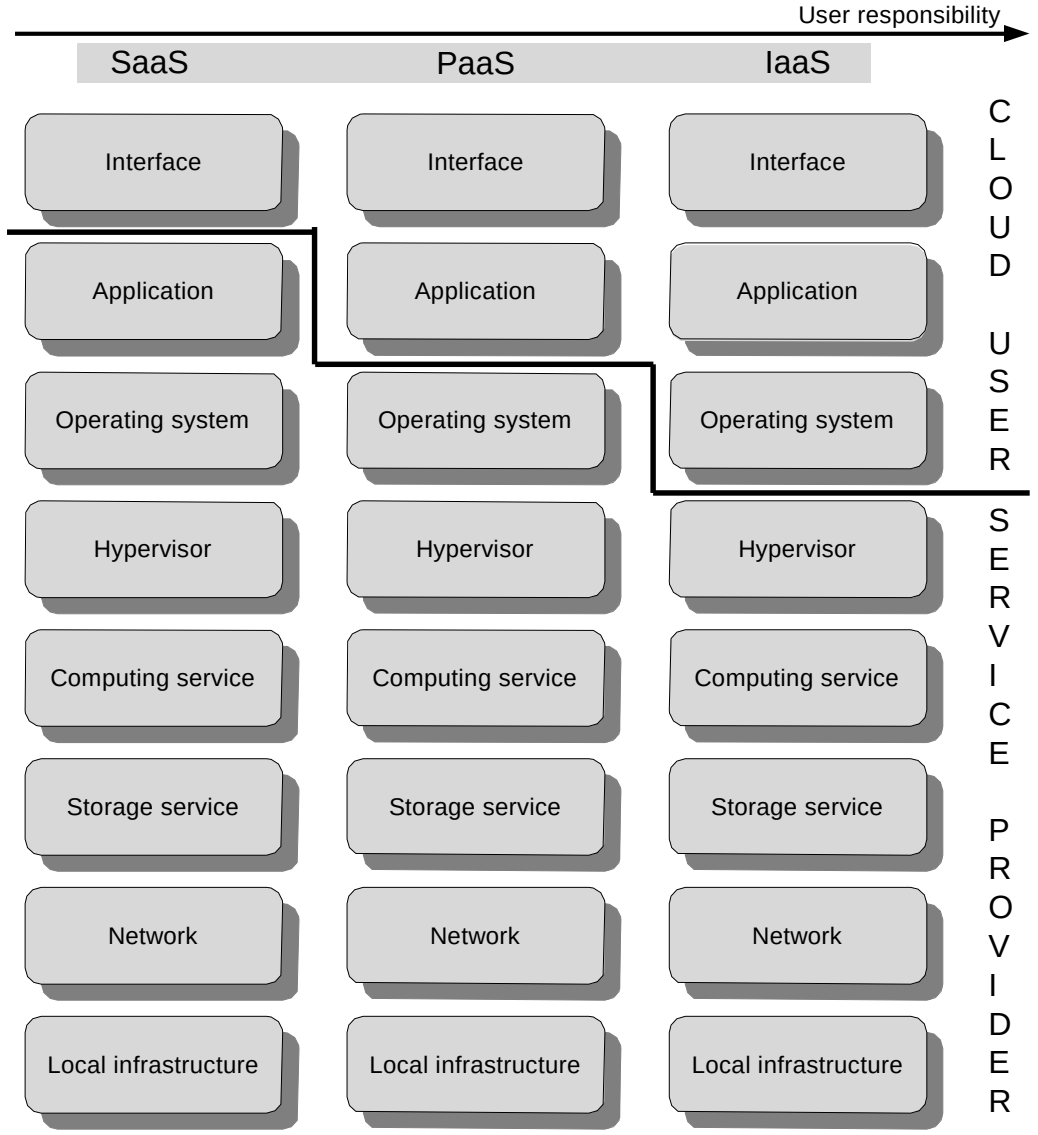
- Virtualization is the process of converting a physical IT resource into a virtual IT resource.
- Most types of IT resources can be virtualized, including:
  - *Servers* - A physical server can be abstracted into a virtual server.
  - *Storage* - A physical storage device can be abstracted into a virtual storage device or a virtual disk.
  - *Network* - Physical routers and switches can be abstracted into logical network fabrics, such as VLANs.
  - *Power* - A physical UPS and power distribution units can be abstracted into what are commonly referred to as virtual UPSs.

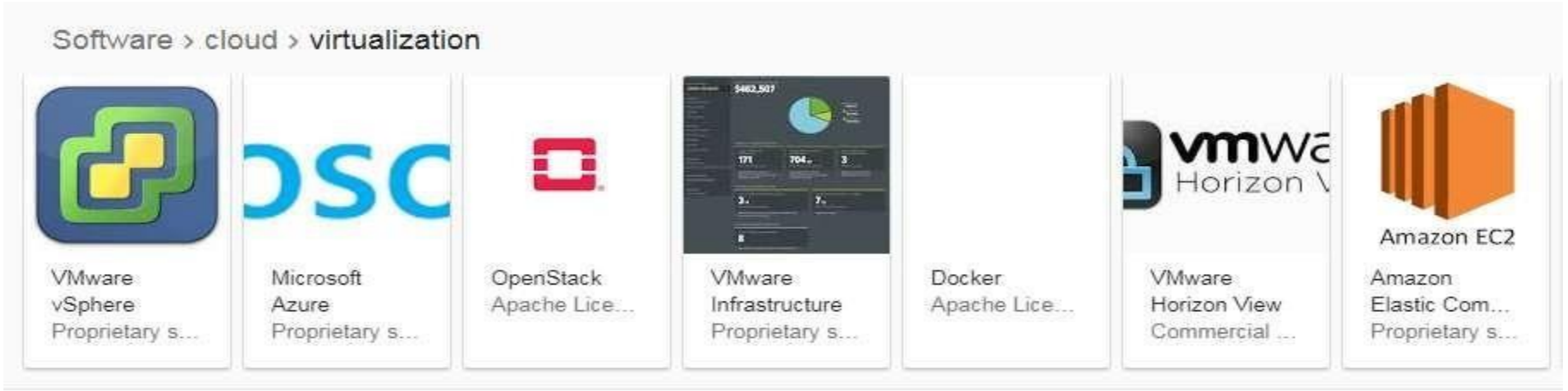
Virtualization Technology

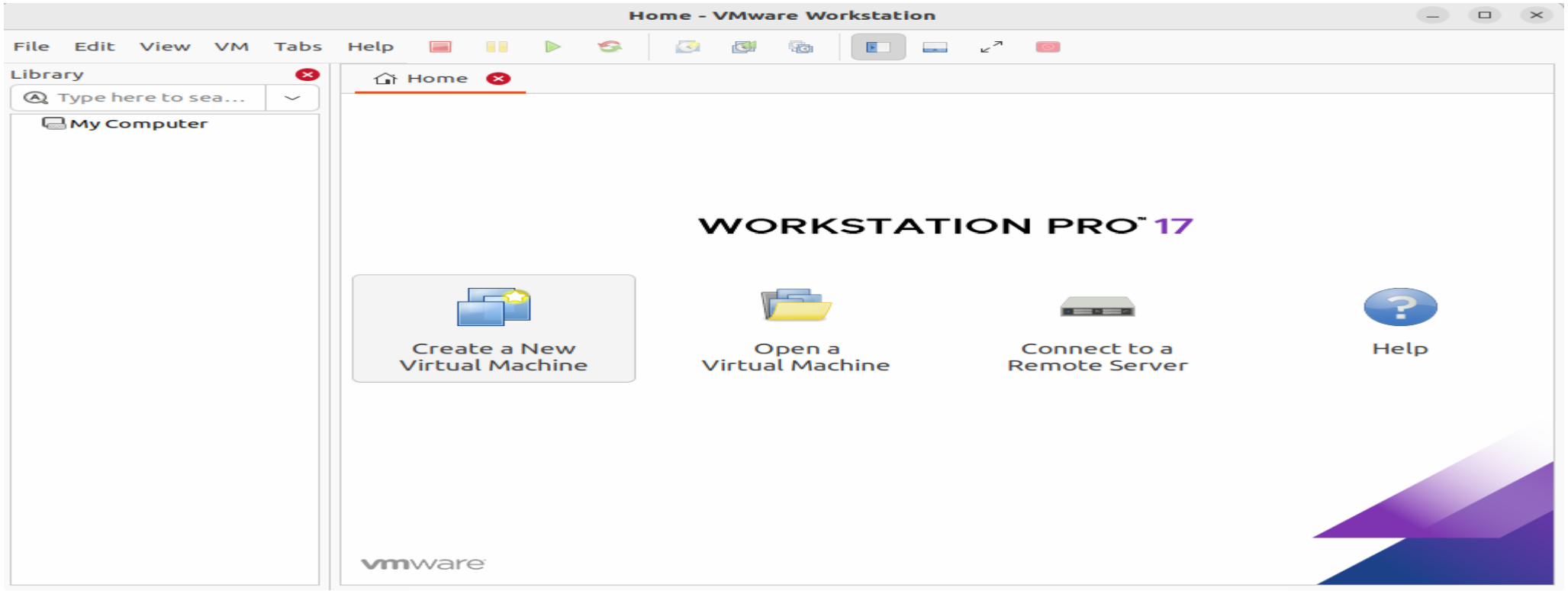
- We can take software and put software on the server in order to get multiple machines on it.
- The software that will be installed is called Hypervisor-it allows to make virtual machines on this physical machine.
- With this software several virtual machines can be put on a single physical machine.

Software allowing us to create virtual machines VMs.

Virtualization and Services

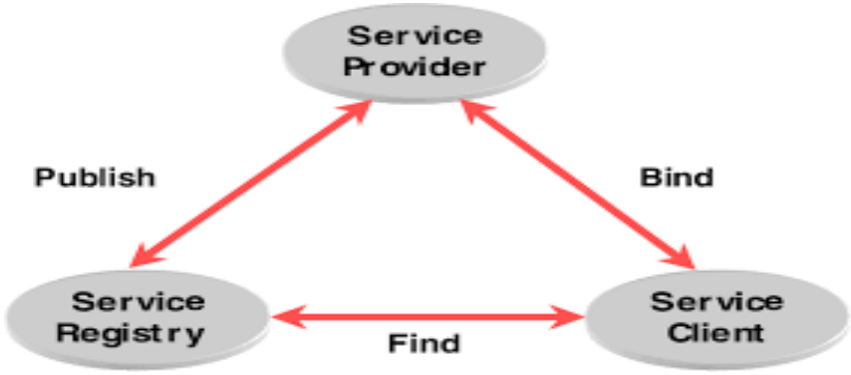



Software › cloud › virtualization
VMware vSphere
Proprietary s...
Microsoft Azure
Proprietary s...
OpenStack
Apache Lice...
VMware Infrastructure
Proprietary s...
Docker
Apache Lice...
VMware Horizon View
Commercial ...
Amazon EC2
Amazon Elastic Com...
Proprietary s...


Home - VMware Workstation
File Edit View VM Tabs Help
Library
Type here to sea...
My Computer
Home
WORKSTATION PRO™ 17
Create a New Virtual Machine
Open a Virtual Machine
Connect to a Remote Server
Help
vmware

Service-Oriented Computing (SOC)

- Web Services and Service Oriented Architecture (SOA) represent the base technologies for cloud computing.
- Cloud services are typically designed as Web services, which follow industry standards including WSDL, SOAP, and UDDI. A Service Oriented Architecture organizes and manages Web services inside clouds. A SOA also includes a set of cloud services, which are available on various distributed platforms.



Concept of Service Oriented Architecture (SOA)

- **SOA** is an architectural design that:

Helps to build or extend modular systems in a flexible and reusable manner.

Provides loose coupling among the modules of an application.

- SOA defines three key roles:

Service Provider (Server)

Service Consumer (Client)

Service Broker (Middleware)

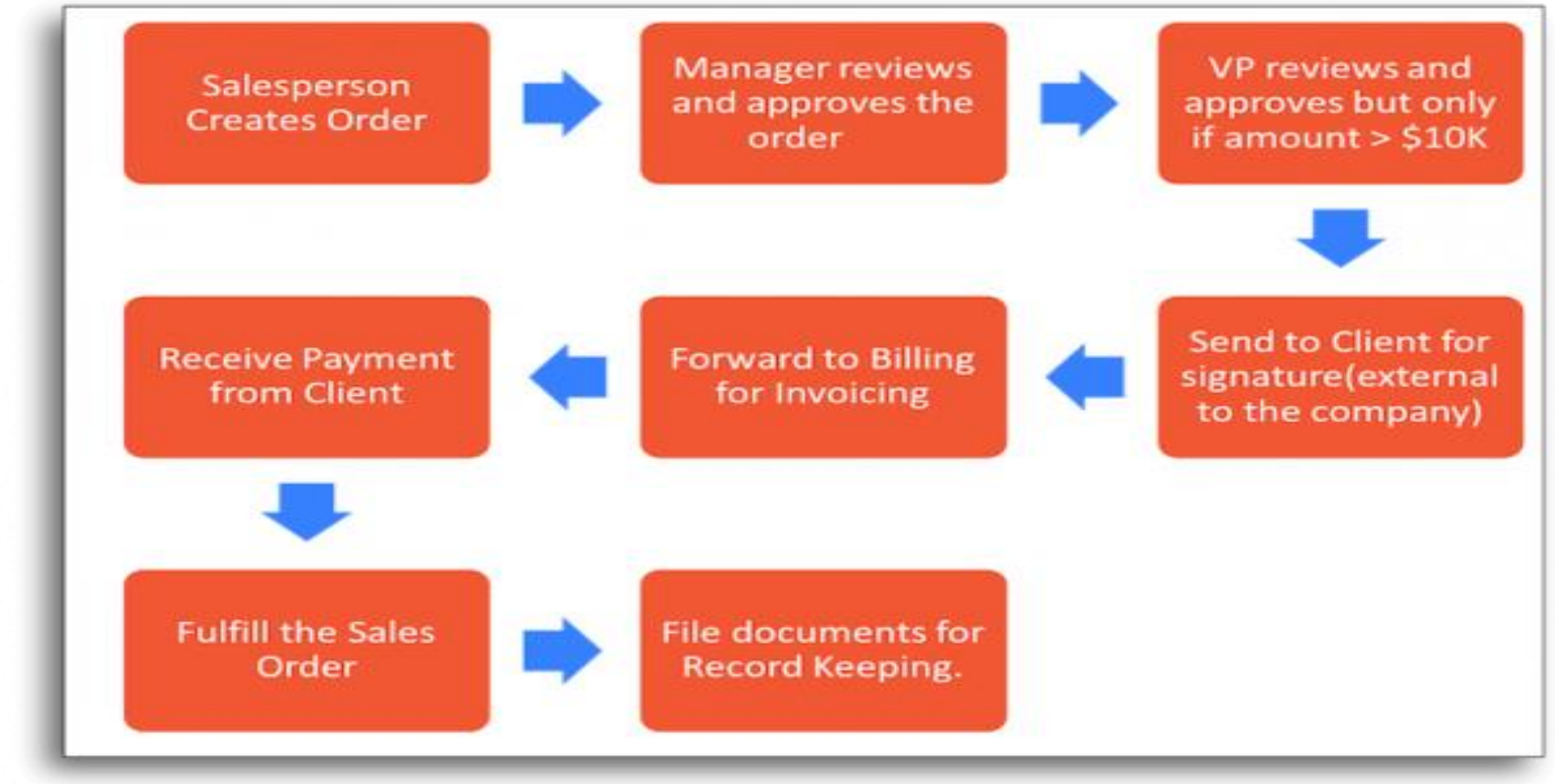
key principles of Service-Oriented Computing

- **Service:** A service is a self-contained unit of functionality that is accessible over a network. It encapsulates a specific business capability and can be invoked by other services or applications.
- **Service-Oriented Architecture (SOA):** SOA is an architectural style that promotes the use of services as the primary building blocks for constructing applications. It involves designing systems based on the principles of loose coupling, service composition, and service discovery.
- **Service Composition:** Service composition involves combining multiple services to create new, more complex services or workflows. It allows the creation of value-added functionality by orchestrating the interactions between different services. Service composition may involve sequential execution, parallel execution, or conditional branching based on business rules.
- **Service Discovery:** Service discovery is the process of locating and binding to available services dynamically. It enables services to be discovered and invoked at runtime without prior knowledge of their locations. Service registries or service directories are commonly used to facilitate service discovery.
- **Service Interoperability:** Interoperability ensures that services can work together seamlessly, regardless of the underlying technologies or platforms they are implemented on. Standards-based protocols and data formats, such as XML, JSON, and Web Services Description Language (WSDL), are used to enable interoperability between services.
- **Service Governance:** Service governance involves managing and controlling the lifecycle of services within an organization. It includes activities such as service registration, versioning, policy enforcement, monitoring, and security. Service governance ensures that services adhere to organizational standards, policies, and quality requirements.

Service Flow and Workflows

- The concept of service flow and workflow refers to an integrated view of service based activities provided in clouds.
- Service flow and workflows are important concepts in cloud computing that help define how cloud services are designed, deployed, and managed.
- In cloud computing, workflows are mostly manages by pipelines that are used to automate and streamline complex processes in cloud computing.

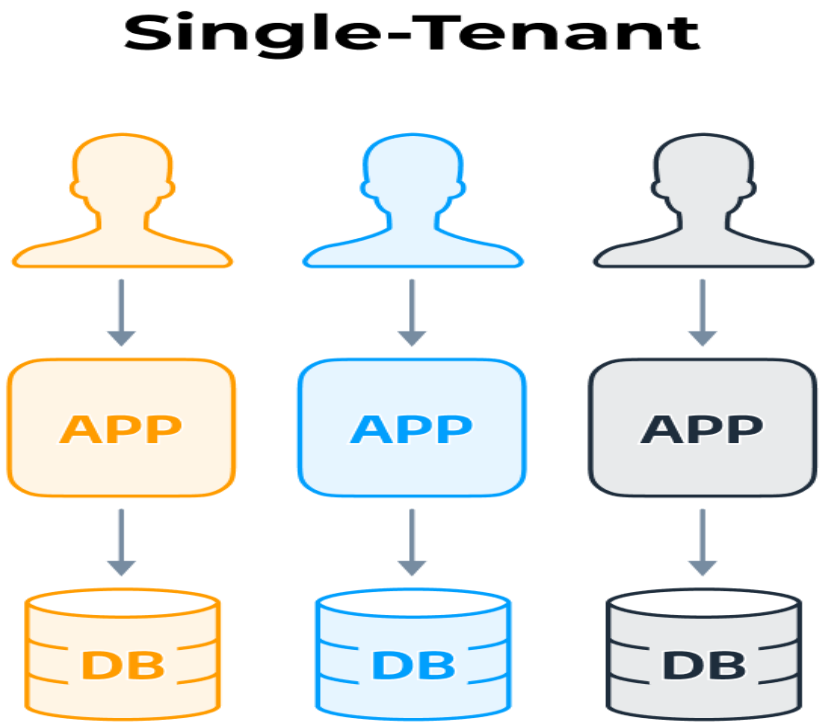
Example - Workflow

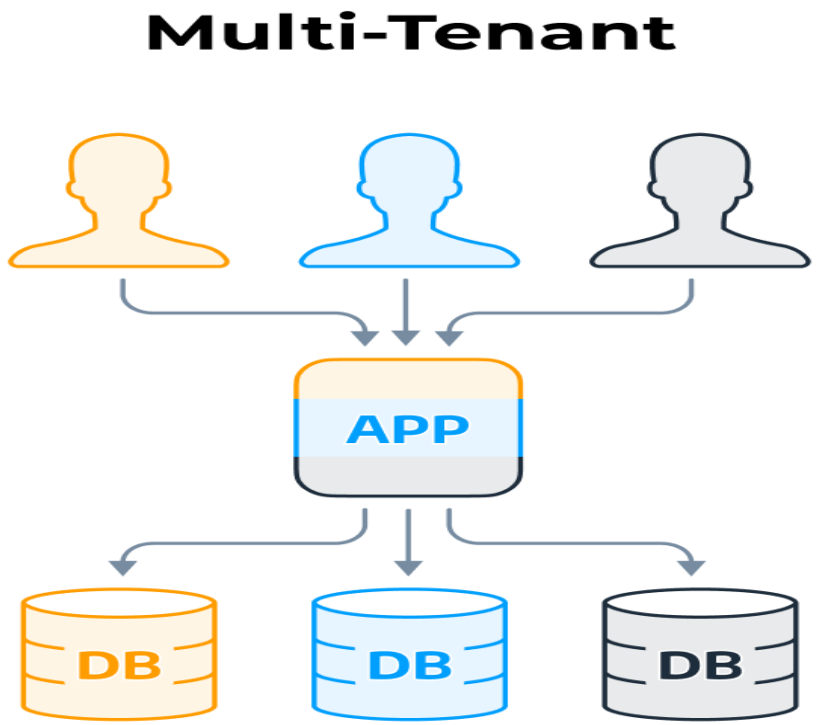


Multitenant technology

- Multitenancy is a software architecture model which means a server runs a single software instance for each tenant yet serves multiple tenants.
- It enable multiple users (tenants) to access the same application simultaneously
- Multitenant applications ensure that tenants do not have access to data and configuration information that is not their own

Single vs Multi-Tenant





Source: Ascendixtech.com

AscendiX

3 Types of Multi Tenancy in Cloud Computing

1. High Degree

This type of multi tenancy means that you can share the database schema and support customized business logic, workflows, and user-interface layers. Simply put, multi tenant software architecture is offered within all the sub-layers of SaaS software.

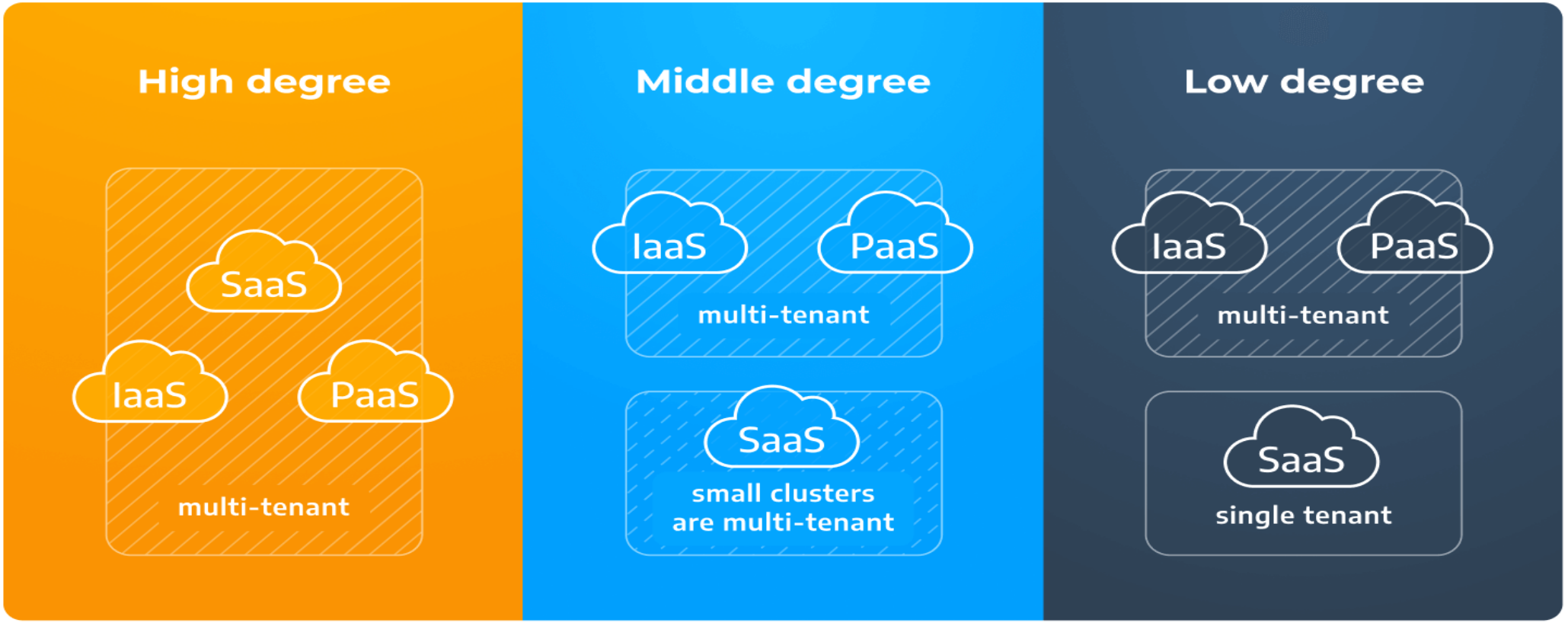
2. Middle Degree

This type of multi tenancy in cloud computing implies that tenants are divided into small clusters that application layers and database schemas. Each tenant cluster has its own copy of the database schema and the application instance. In general, a middle degree means that only small SaaS clusters are multi-tenant.

3. Low Degree

This degree's multi tenant software architecture is available for IaaS and PaaS layers providing dedicated SaaS layers for each tenant.

3 Types of Multi Tenancy

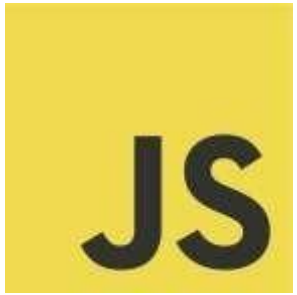


Source: Ascendixtech.com

Ascendix

Web 2.0

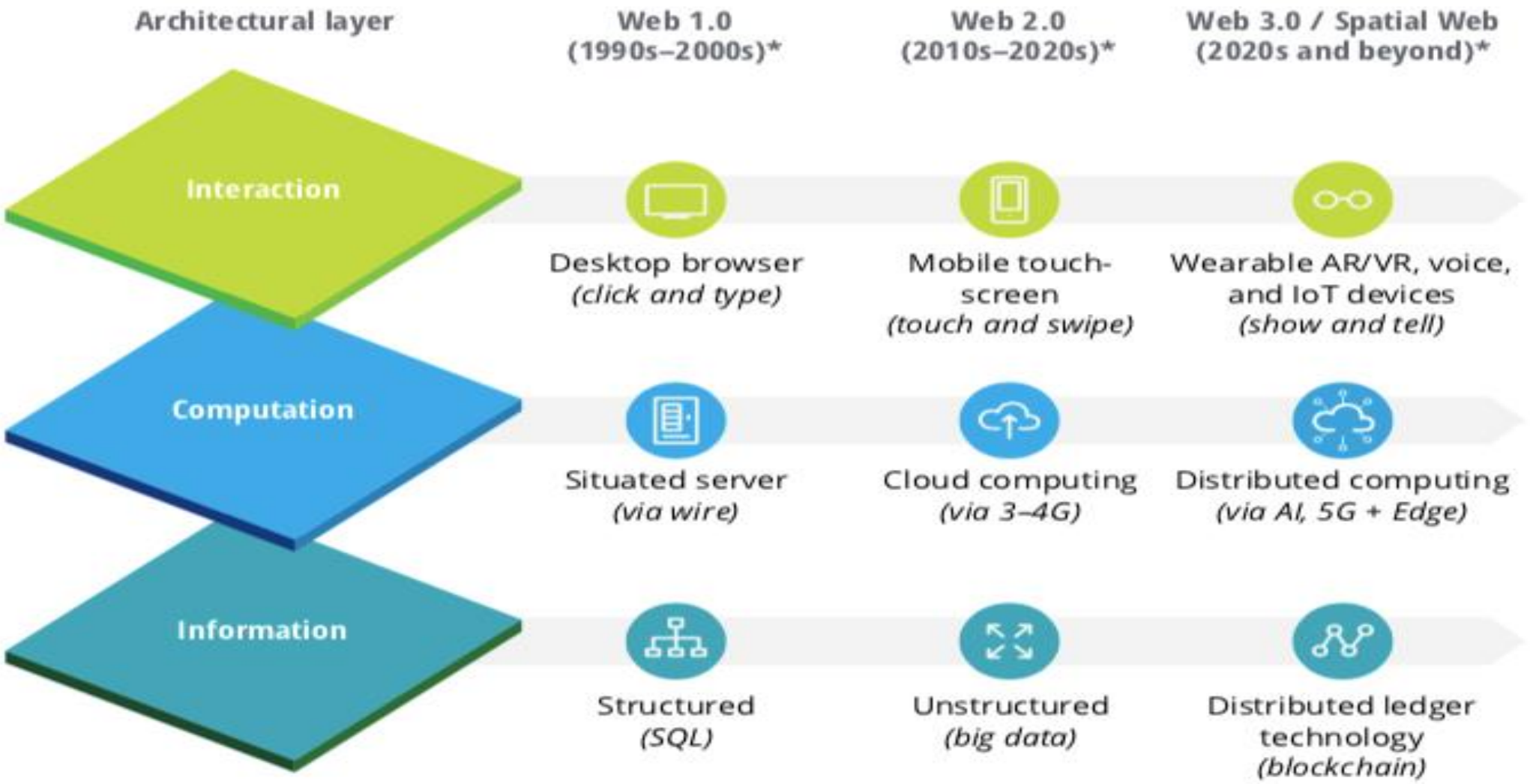
- The second stage of development of the Internet, characterized especially by the change from static web pages to dynamic or user-generated content and the growth of social media.




twitter
Linked in
facebook
zynga
WIKIPEDIA
hulu
tumblr.
You Tube
ZOHO
Work. Online
flickr
Google maps
NETFLIX
PANDORA
yelp
Google I/O

Web 3.0

- Web 3.0 also known as Web 3, is a concept for the next generation of the World Wide Web.
- The vision for Web 3.0 is that it will be a version of the internet that is both open and decentralized.
- Decentralization means that the internet isn’t controlled by a single body like the government but by a community of members. It is millions of devices linked together in an open network.


Architectural layer
Web 1.0 (1990s–2000s)*
Web 2.0 (2010s–2020s)*
Web 3.0 / Spatial Web (2020s and beyond)*
Interaction
Desktop browser (click and type)
Mobile touch-screen (touch and swipe)
Wearable AR/VR, voice, and IoT devices (show and tell)
Computation
Situated server (via wire)
Cloud computing (via 3–4G)
Distributed computing (via AI, 5G + Edge)
Information
Structured (SQL)
Unstructured (big data)
Distributed ledger technology (blockchain)

Web 1.0

100,000 websites

(read-only Web)

published content

user-generated content

50,000,000 users

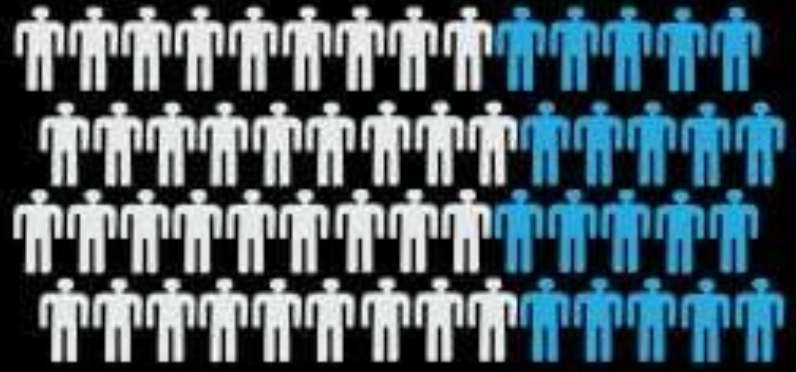
Web 2.0

100,000,000 websites

(read-write Web)

published content

user-generated content



1,000,000,000 users

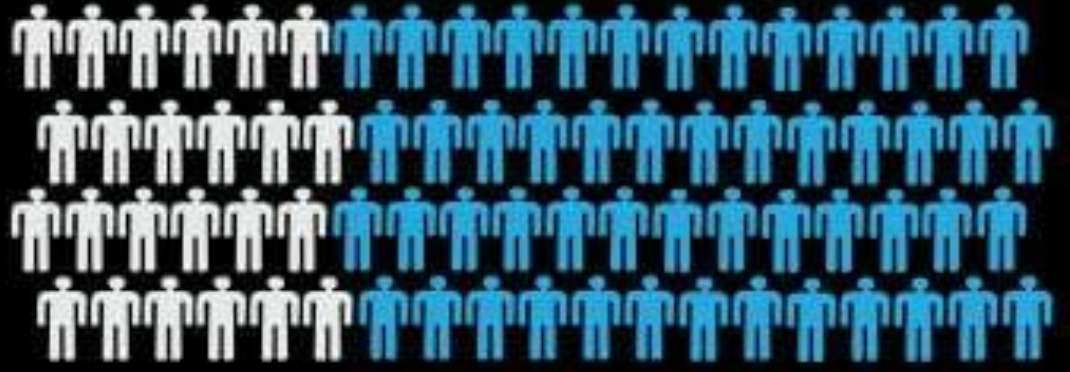
Web 3.0

1,000,000,000 websites

(read-write Web)

published content

user-generated content



2,500,000,000 users

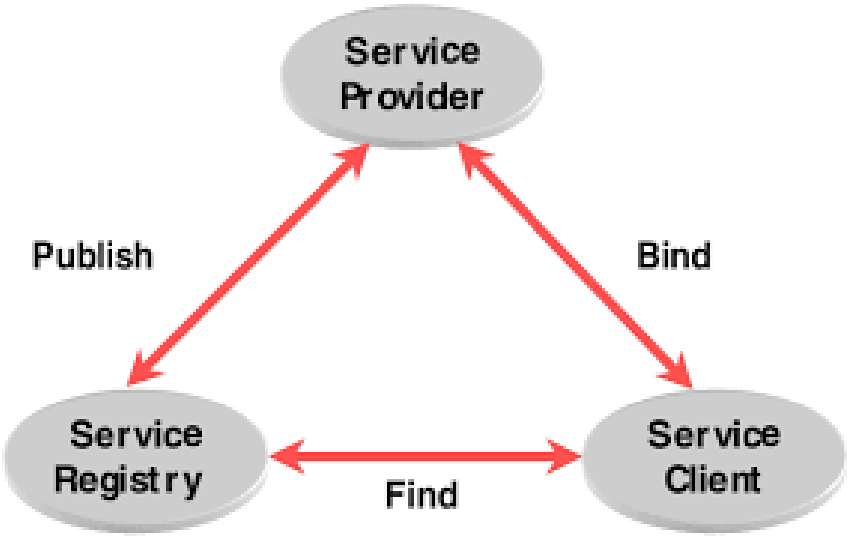
Service-Oriented Computing (SOC)

components into a network of services to create applications.

- Uses "services-oriented" programming to develop application by using network-available services.
- Web services are currently the most promising SOC-based technology. Uses internet-based standards:
  - Simple Object Access Protocol (SOAP)
  - Web Services Description Language (WSDL)
  - Business Process Execution Language for Web Services (BPEL4WS)

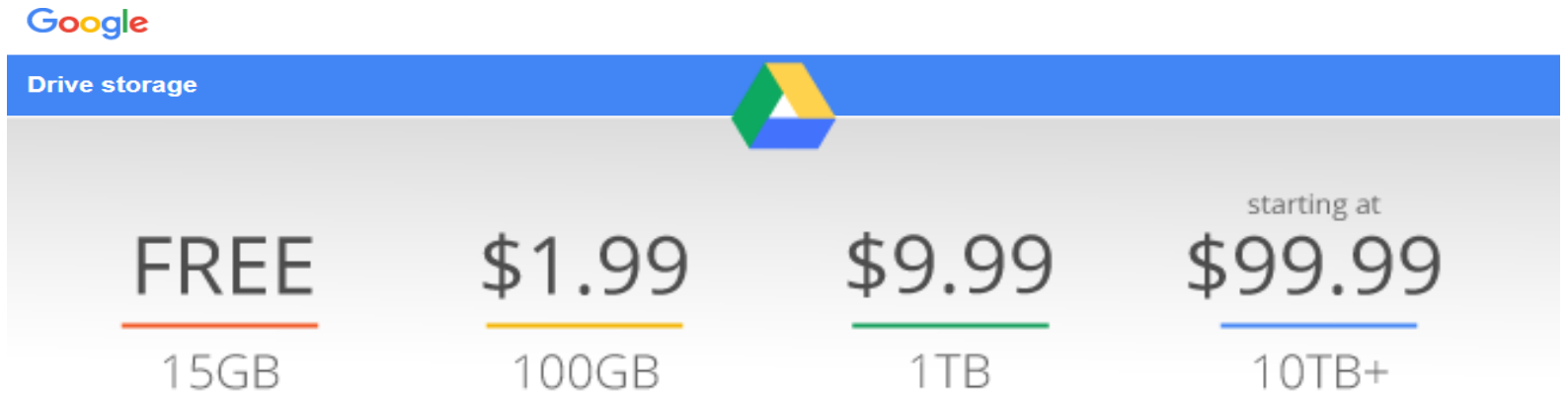
Service Oriented Computing

- Core Reference model for Cloud Computing System
- SOC Introduce Two main Concepts
  - Quality of Service (QOS)
  - Software as Service (SaaS)



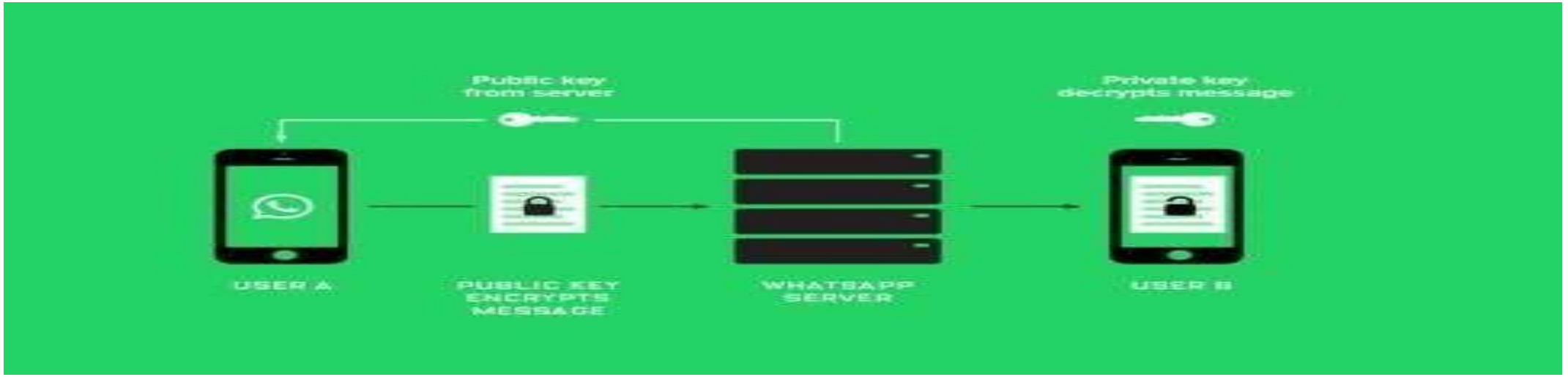
Utility Oriented Computing

- **The Computer Utility**, is a service provisioning model in which a service provider makes computing resources and infrastructure management available to the customer as needed, and charges them for specific usage rather than a flat rate.



Building Cloud Computing Environment

- Application Development



Enterprise Application

Infrastructure and System Development