

# Cloud Computing

## Unit 01



# Types of Clouds

- Cloud computing is offered in different forms:
  - ✓ Public clouds
  - ✓ Private clouds
  - ✓ Hybrid clouds, which combine both public and private

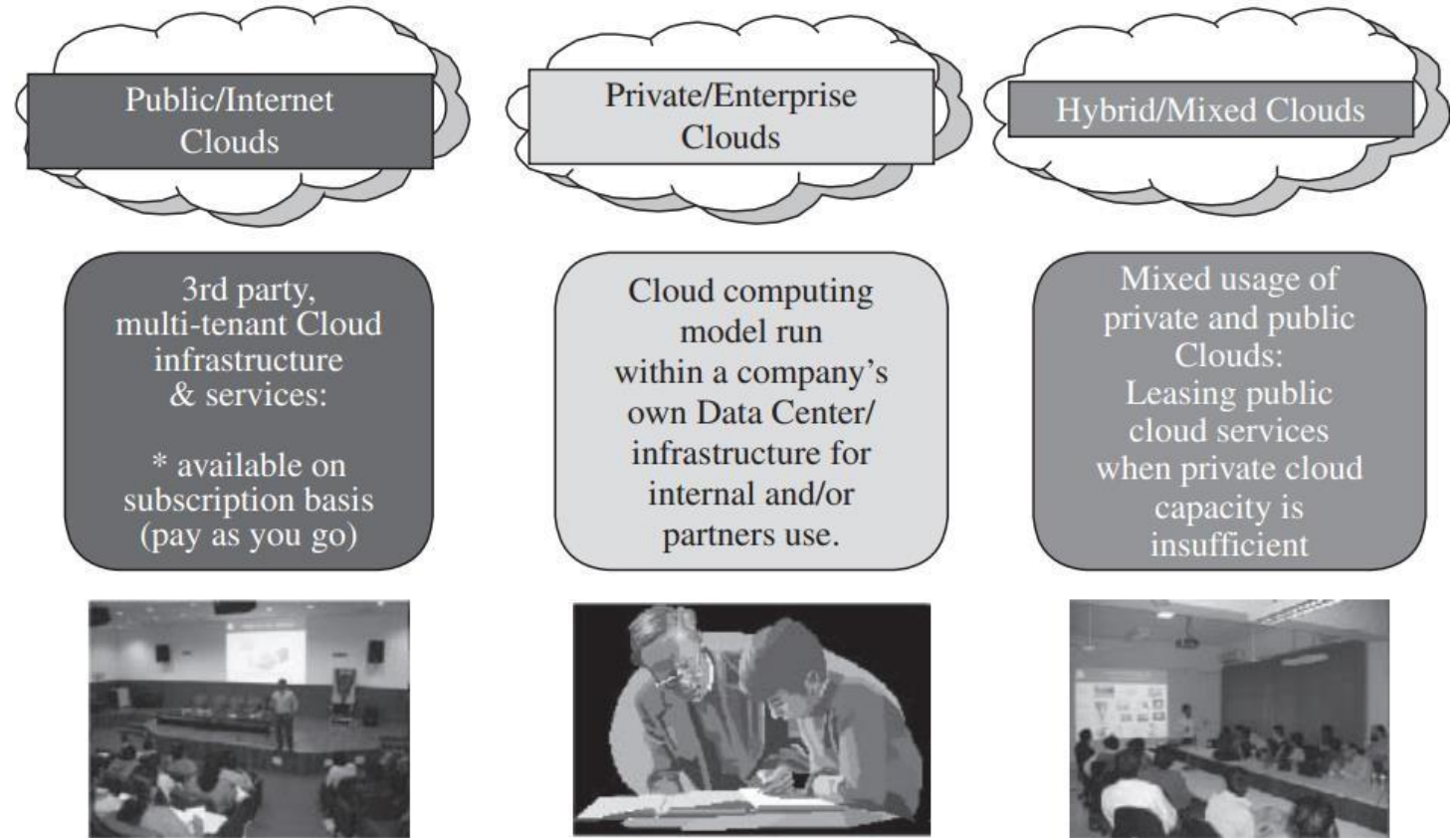





FIGURE 1.4. Types of clouds based on deployment models

# Layers of Clouds

- Cloud computing services are divided into three classes, according to the abstraction level of the capability provided and the service model of providers, namely:

- (1) Infrastructure as a Service
- (2) Platform as a Service
- (3) Software as a Service

Service Class	Main Access & Management Tool	Service content
 SaaS	Web Browser	<b>Cloud Applications</b> Social networks, Office suites, CRM, Video processing
 PaaS	Cloud Development Environment	<b>Cloud Platform</b> Programming languages, Frameworks, Mashups editors, Structured data
 IaaS	Virtual Infrastructure Manager	<b>Cloud Infrastructure</b> Compute Servers, Data Storage, Firewall, Load Balancer

[Types of Cloud Computing - SaaS vs PaaS vs IaaS - AWS \(amazon.com\)](#)

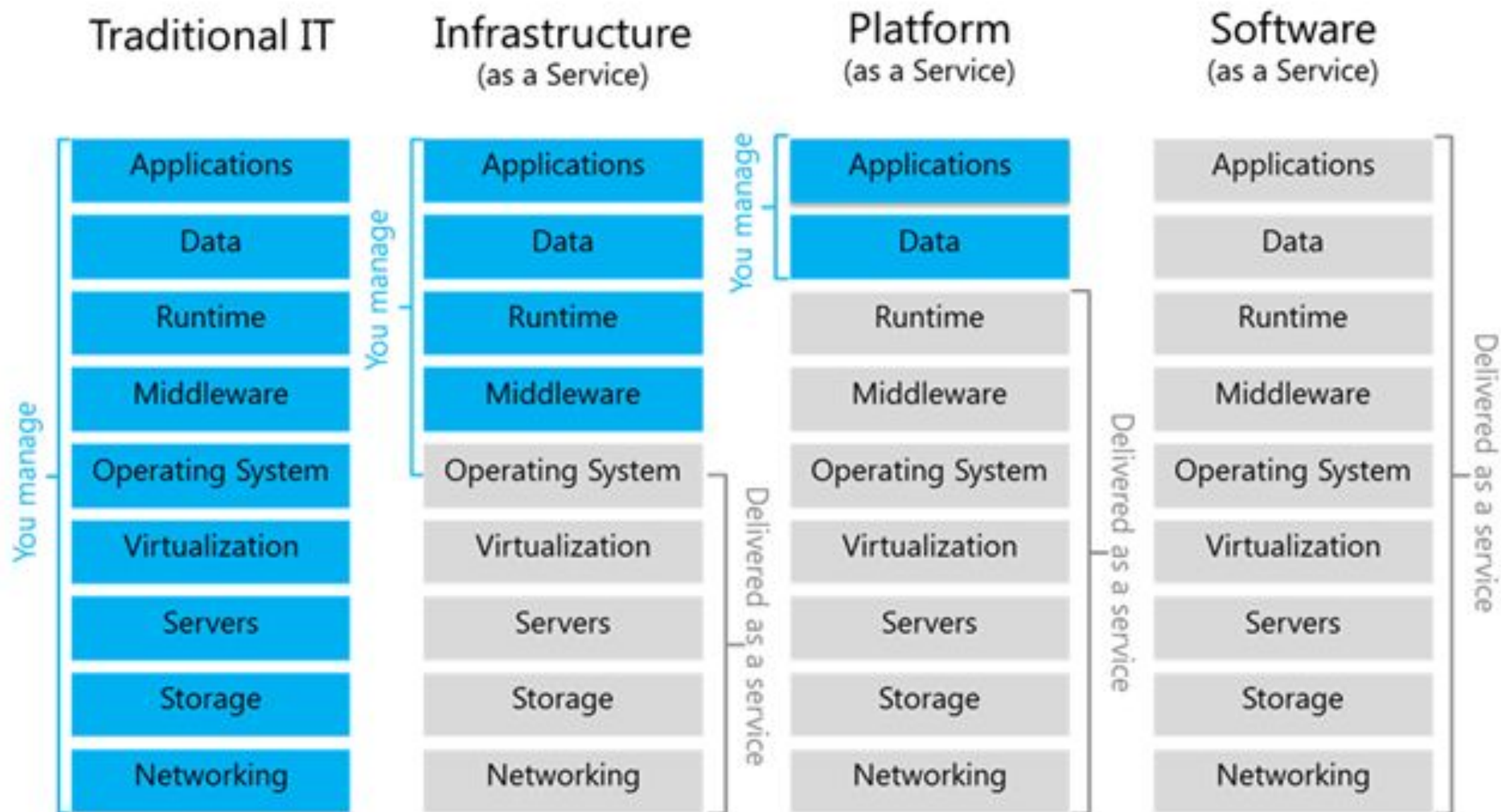
[PaaS - Azure Services Overview \(azurecharts.com\)](#)

[PaaS vs IaaS vs SaaS: What's the difference? | Google Cloud](#)

[What is IaaS? Infrastructure as a Service | Microsoft Azure](#)

FIGURE 1.3. The cloud computing stack

# Layers of Clouds



## Layers of Clouds: IaaS

- ✓ **The Infrastructure as a Service** layer offers storage and computer resources that developers and IT organizations use to deliver custom business solutions.

Offering virtualized resources

- computation,
- storage
- communication
- 

**on demand** is known as Infrastructure as a Service (IaaS)

Infrastructure services are considered to be the **bottom layer of cloud computing** systems [39].

## Layers of Clouds: PaaS

- ✓ **The Platform as a Service** layer offers development environments that IT organizations can use to create cloud-ready business applications.

Provider **delivers more than infrastructure.**

It delivers what you might call a solution stack — an **integrated set of software** that provides everything a developer needs **to build an application** — for both software development and runtime.

PaaS can be viewed as an evolution of Web hosting.

The primary benefit of PaaS is having software development and deployment capability based entirely in the cloud — hence, **no management or maintenance efforts are required for the infrastructure.**

# Layers of Clouds: SaaS

- ✓ **The Software as a Service** layer offers purpose-built business applications.

Model of delivering applications, known as Software as a Service (SaaS), alleviates the burden of software maintenance for customers and simplifies development and testing for providers.

Software as a Service comes in two distinct modes:

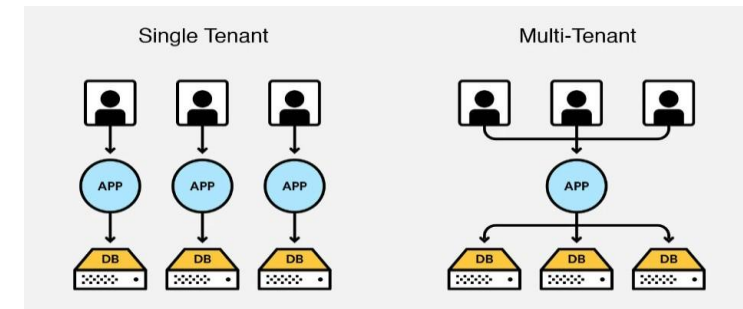
- ✓ **Simple multi-tenancy:**

Each customer has its own resources that are segregated from those of other customers. It amounts to a **relatively inefficient** form of multi-tenancy.

- ✓ **Fine grain multi-tenancy:**

This offers the same level of segregation but from a software engineering perspective, it's far more efficient.

All resources are shared, but **customer data and access capabilities are segregated** within the application. This offers much superior economies of scale



## Layers of Clouds:

1. Infrastructure as a Service (IaaS): IaaS offers virtualized computing resources (servers, storage, networking) on-demand. Users can scale resources up or down as needed. [Examples include Amazon EC2, Microsoft Azure VMs, and Google Compute Engine1.](#)
2. Platform as a Service (PaaS): PaaS provides a platform for developers to build, deploy, and manage applications. It abstracts infrastructure complexities, allowing developers to focus on coding. [Examples include Google App Engine, Microsoft Azure App Service, and Heroku2.](#)
3. Software as a Service (SaaS): SaaS delivers software applications over the internet. Users access them via a web browser, eliminating the need for local installations. [Examples include Google Workspace, Microsoft 365, and Salesforce1.](#)



## Challenges and Risks:

1. **Vendor Dependency:** Organizations relying heavily on cloud services may become dependent on specific vendors.
2. **Security Risks:** Data breaches, unauthorized access, and vulnerabilities are concerns.
3. **Compatibility with Existing Infrastructure:** Migrating to the cloud can be complex due to compatibility issues.
4. **Cost Management:** Cloud services can be cost-effective, but unexpected expenses may arise.

# Cloud Computing

Cloud Computing Architecture,

Deployment Models,

Virtualization,

XML Basics,

Web

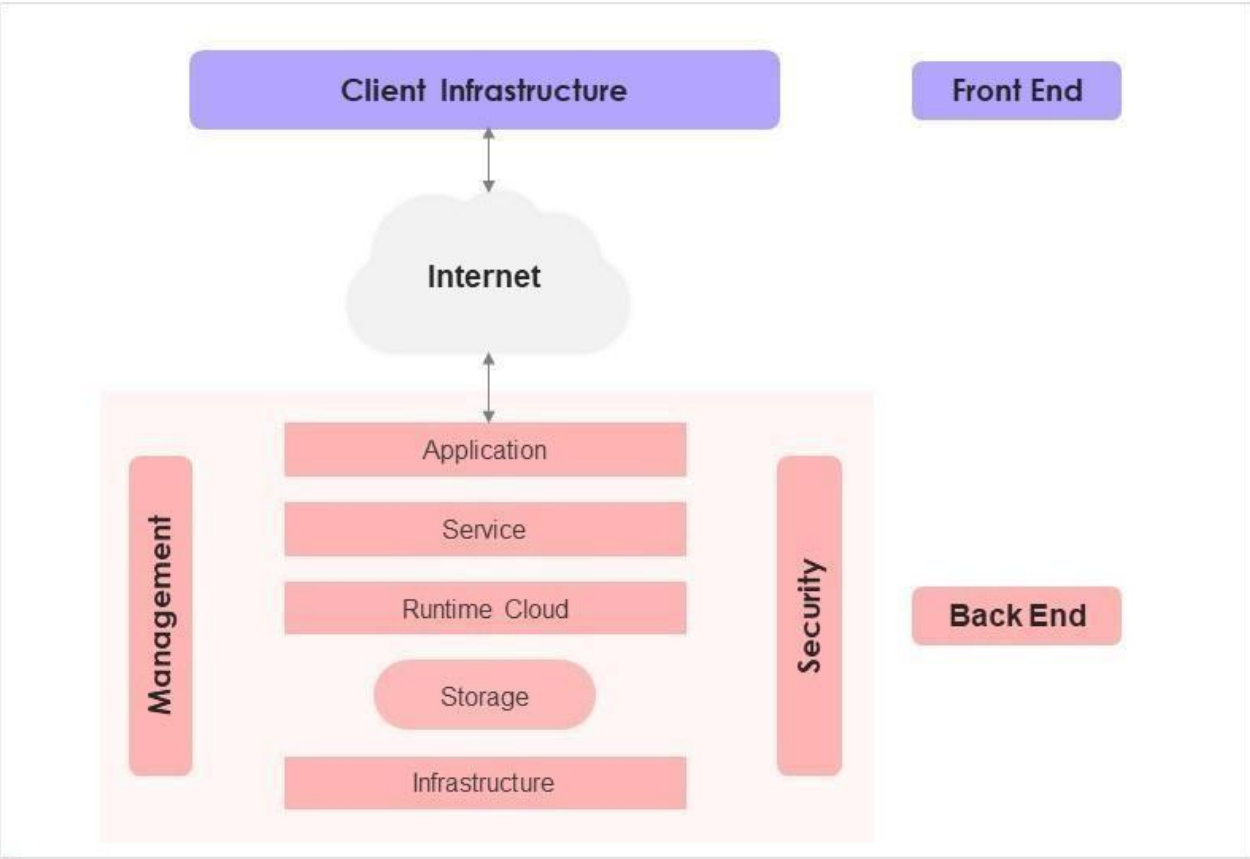
Services,

Service Oriented Architecture

# Cloud Computing Architecture

## Architecture of Cloud Computing

This slide shows cloud computing architecture, including client infrastructure that falls under the front end and back end elements such as application, service, runtime cloud, management, storage, etc.



### Front End

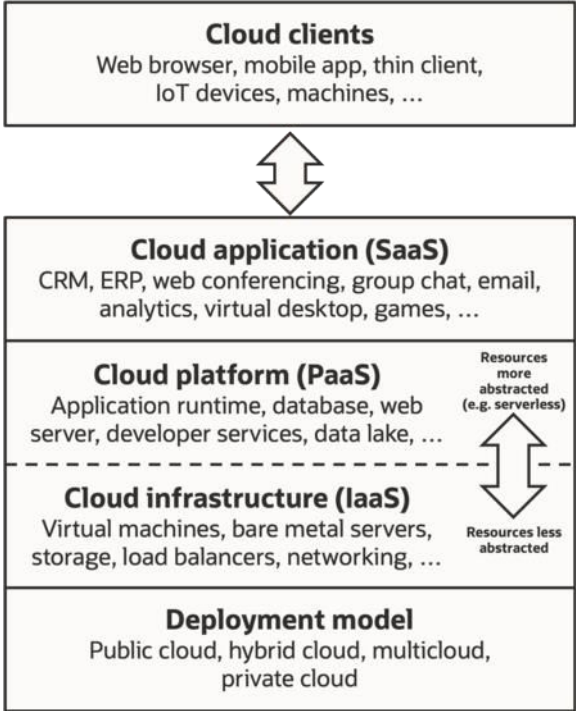
- Clients communicate with the front end
- Includes client-side interfaces and programs for using cloud computing systems
- Web browsers, thin and fat clients, tablets, and mobile devices are all part of the front end

### Back End

- Service provider utilizes the back end
- Oversees all of the resources needed to deliver cloud computing services
- Comprises a massive quantity of data storage and security methods, virtual machines, deployment models, servers, and traffic management systems, among other things

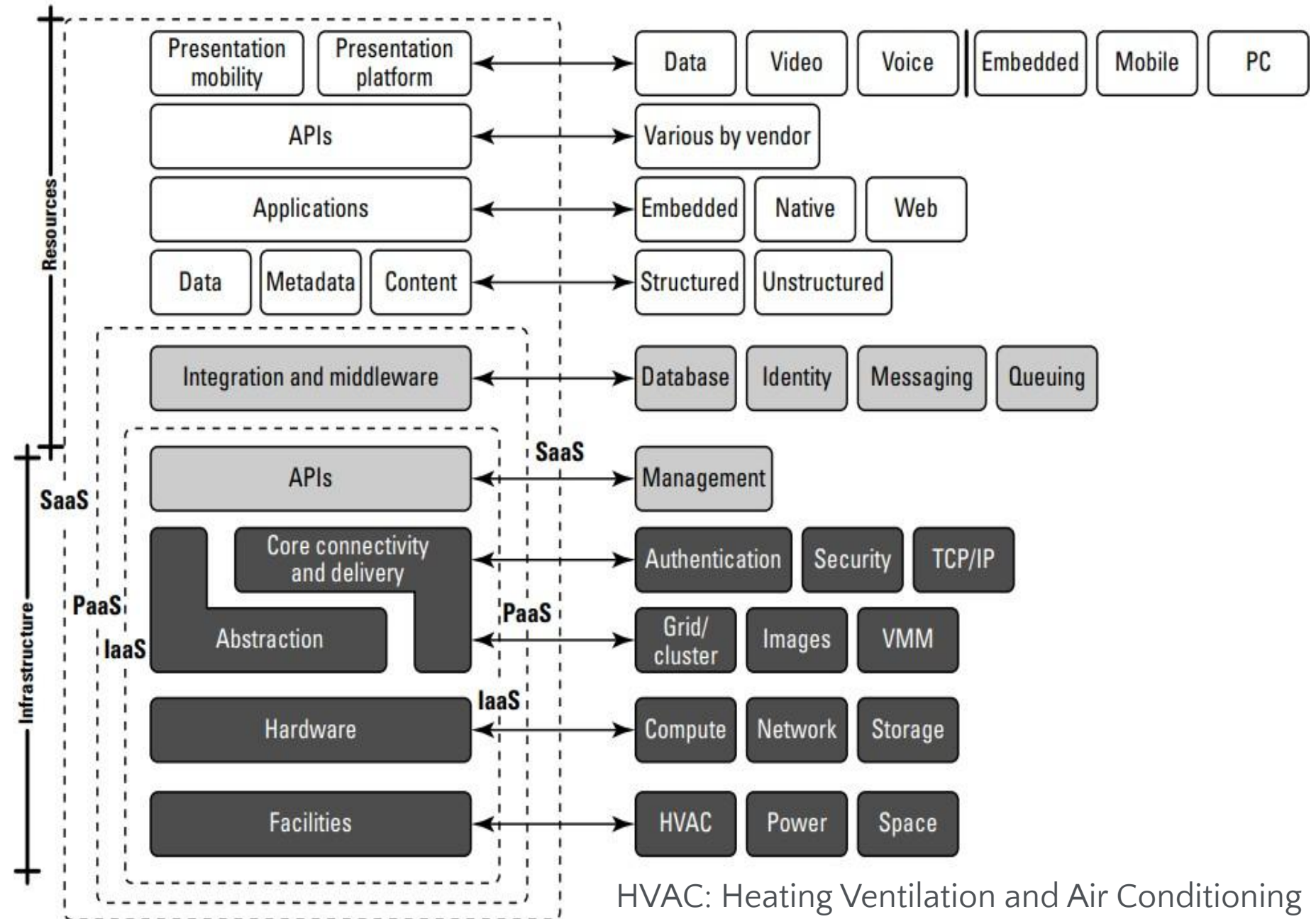
### Add text here

- Add text here
- Add text here



# Cloud Computing Architecture

## The Cloud Reference Model



# *Cloud Computing Architecture*

A simpler way of understanding how cloud architecture works is to think of all these components as various layers placed on top of each other to create a cloud platform.

Here are the basic cloud architecture layers:

## **1. Hardware:**

The servers, storage, network devices, and other hardware that power the cloud.

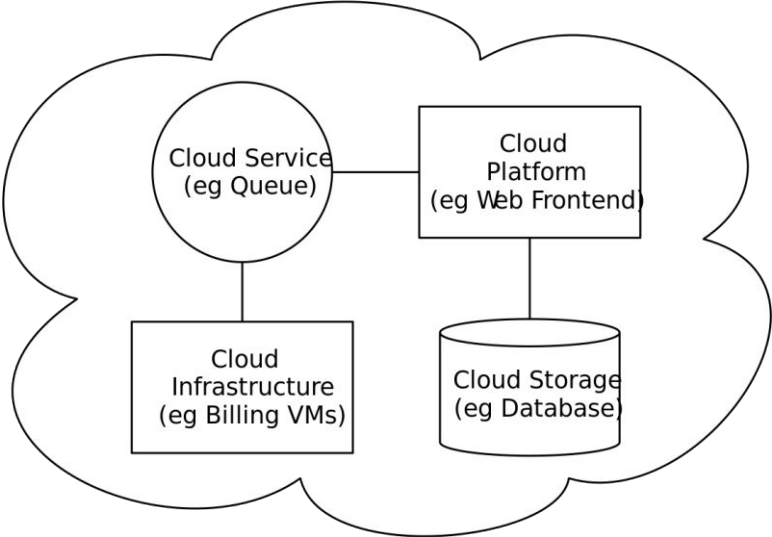
## **2. Virtualization:**

An abstraction layer that creates a virtual representation of physical computing and storage resources. This allows multiple applications to use the same resources.

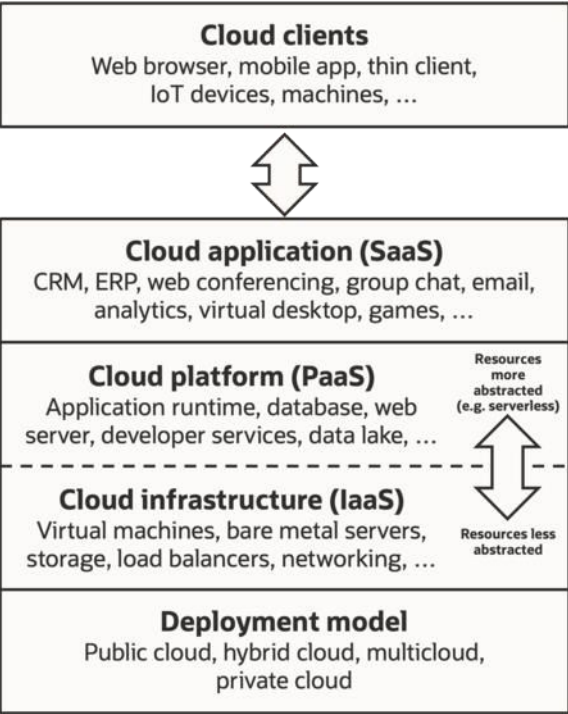
## **3. Application and service:**

This layer coordinates and supports requests from the frontend user interface, offering different services based on the cloud service model,

# Cloud Computing Architecture



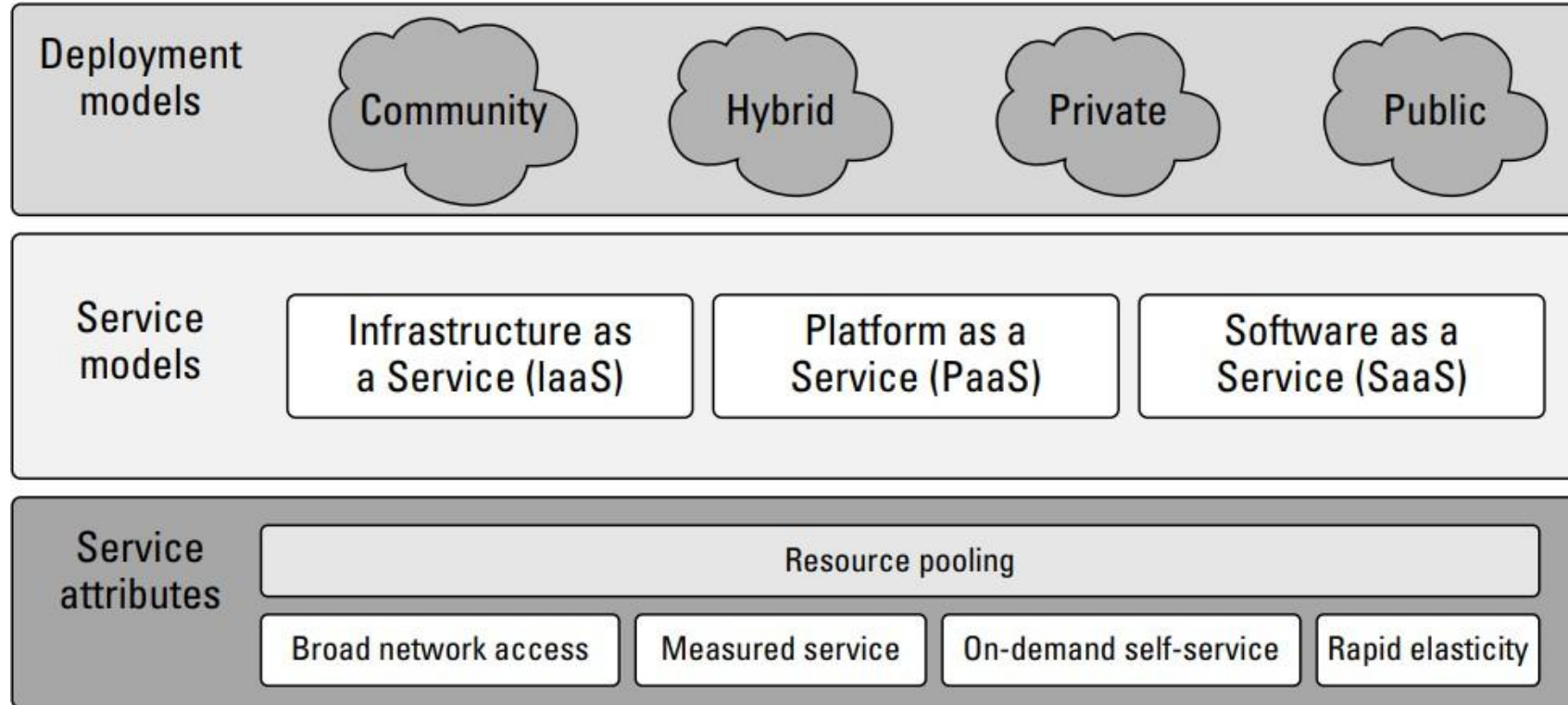
Cloud computing sample architecture



Cloud computing service models arranged as layers in a stack

# Cloud Computing Architecture

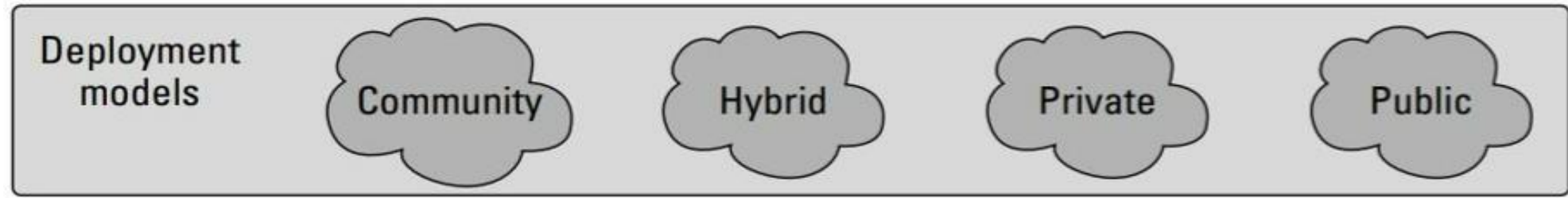
The NIST cloud computing definitions



[Companies Using Salesforce Community Cloud, Market Share, Customers and Competitors \(hgdata.com\)](http://hgdata.com)

[5 Real-World Examples of Cloud Computing Services \(maropost.com\)](http://maropost.com)

## Cloud Computing Deployment Models



### The cloud computing service offering and deployment models

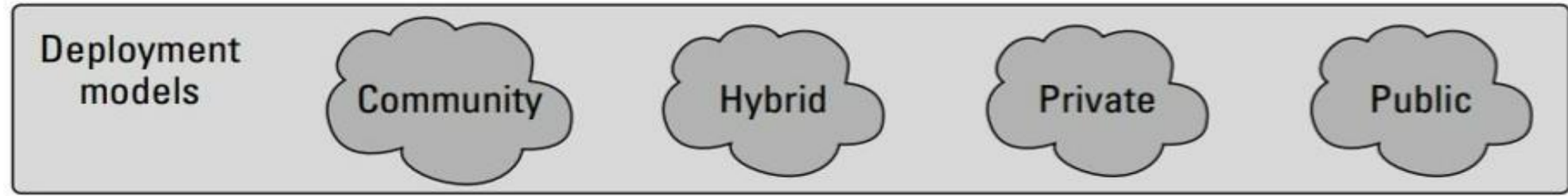
Deployment models A deployment model defines the purpose of the cloud and the nature of how the cloud is located. The NIST definition for the four deployment models is as follows:

**Public cloud:** The public cloud infrastructure is available for public use alternatively for a large industry group and is owned by an organization selling cloud services.

**Private cloud:** The private cloud infrastructure is operated for the exclusive use of an organization. The cloud may be managed by that organization or a third party. Private clouds may be either on- or off-premises.



## Cloud Computing Deployment Models



The cloud computing service offering and deployment models

**Hybrid cloud:** A hybrid cloud combines multiple clouds (private, community or public) where those clouds retain their unique identities, but are bound together as a unit. A hybrid cloud may offer standardized or proprietary access to data and applications, as well as application portability.

**Community cloud:** A community cloud is one where the cloud has been organized to serve a common function or purpose. It may be for one organization or for several organizations, but they share common concerns such as their mission, policies, security, regulatory compliance needs, and so on. A community cloud may be managed by the constituent organization(s) or by a third party.

# Cloud Computing Virtualization

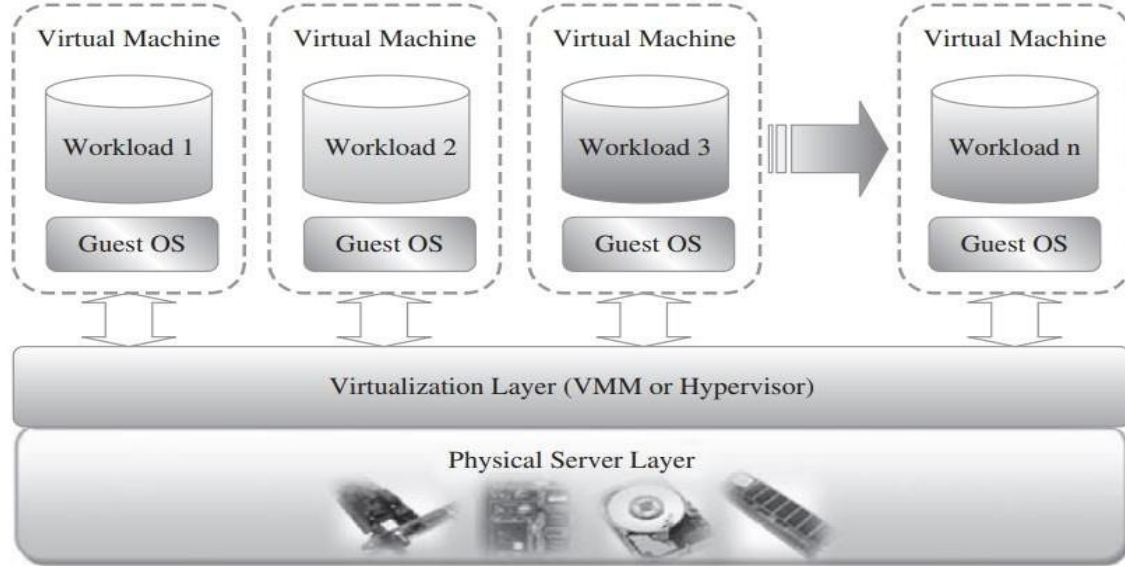


FIGURE A layered virtualization technology architecture.

- **Virtualization:**

Cloud computing virtualizes systems by pooling and sharing resources.

Systems and storage can be provisioned as needed from a centralized infrastructure, costs are assessed on a metered basis, multi-tenancy is enabled, and resources are scalable with quickly

# Cloud Computing Virtualization

- When you use cloud computing, you are **accessing pooled resources** using a technique called virtualization.
- Virtualization **assigns a logical name for a physical resource** and then **provides a pointer to that physical resource** when a request is made.
- Virtualization provides a means to manage resources efficiently because the mapping of virtual resources to physical resources can be both dynamic.
- Virtualization is dynamic in that the mapping can be assigned based on rapidly changing conditions, and it is easy because changes to a mapping assignment can be without any delay.

# Cloud Computing Virtualization characteristic

- **Access:** A client can request access to a cloud service from any location.
- **Application:** A cloud has multiple application instances and directs requests to an instance based on conditions.
- **CPU:** Computers can be partitioned into a set of virtual machines with each machine being assigned a workload. Alternatively, systems can be virtualized through load-balancing technologies.
- **Storage:** Data is stored across storage devices and often replicated for redundancy

# Cloud Computing Virtualization Mobility Patterns

To enable these characteristics, resources must be highly **configurable** and **flexible**.

The features in software and hardware that enable this flexibility as conforming to one or more of the following **mobility patterns**:

- P2V: Physical to Virtual
- V2V: Virtual to Virtual
- V2P: Virtual to Physical
- P2P: Physical to Physical
- D2C: Datacenter to Cloud
- C2C: Cloud to Cloud
- C2D: Cloud to Datacenter
- D2D: Datacenter to Datacenter

# Cloud Computing Virtualization Attributes

- **Service-based:** A service-based architecture is where clients are abstracted from service providers through service interfaces.
- **Scalable and elastic:** Services can be altered to affect capacity and performance on demand.
- **Shared services:** Resources are pooled in order to create greater efficiencies.
- **Metered usage:** Services are billed on a usage basis.
- **Internet delivery:** The services provided by cloud computing are based on Internet protocols and formats.

# Virtual Machine Monitor (VMM) or Hypervisor

- **VMWare**

VMware is a pioneer in the virtualization market.

It is a bare-metal hypervisor, meaning that it installs directly on the physical server, whereas others may require a host operating system.



- **Xen**

Currently forms the **base of commercial hypervisors** like Citrix XenServer [26] and Oracle VM [27].

Xen hypervisor started as an open-source project and has served as a base to other virtualization products, both commercial and open-source.

It has pioneered the **para-virtualization** concept.



- **KVM**

Kernel-based Virtual MachineThe kernel-based virtual machine (KVM) is a **Linux virtualization** subsystem.

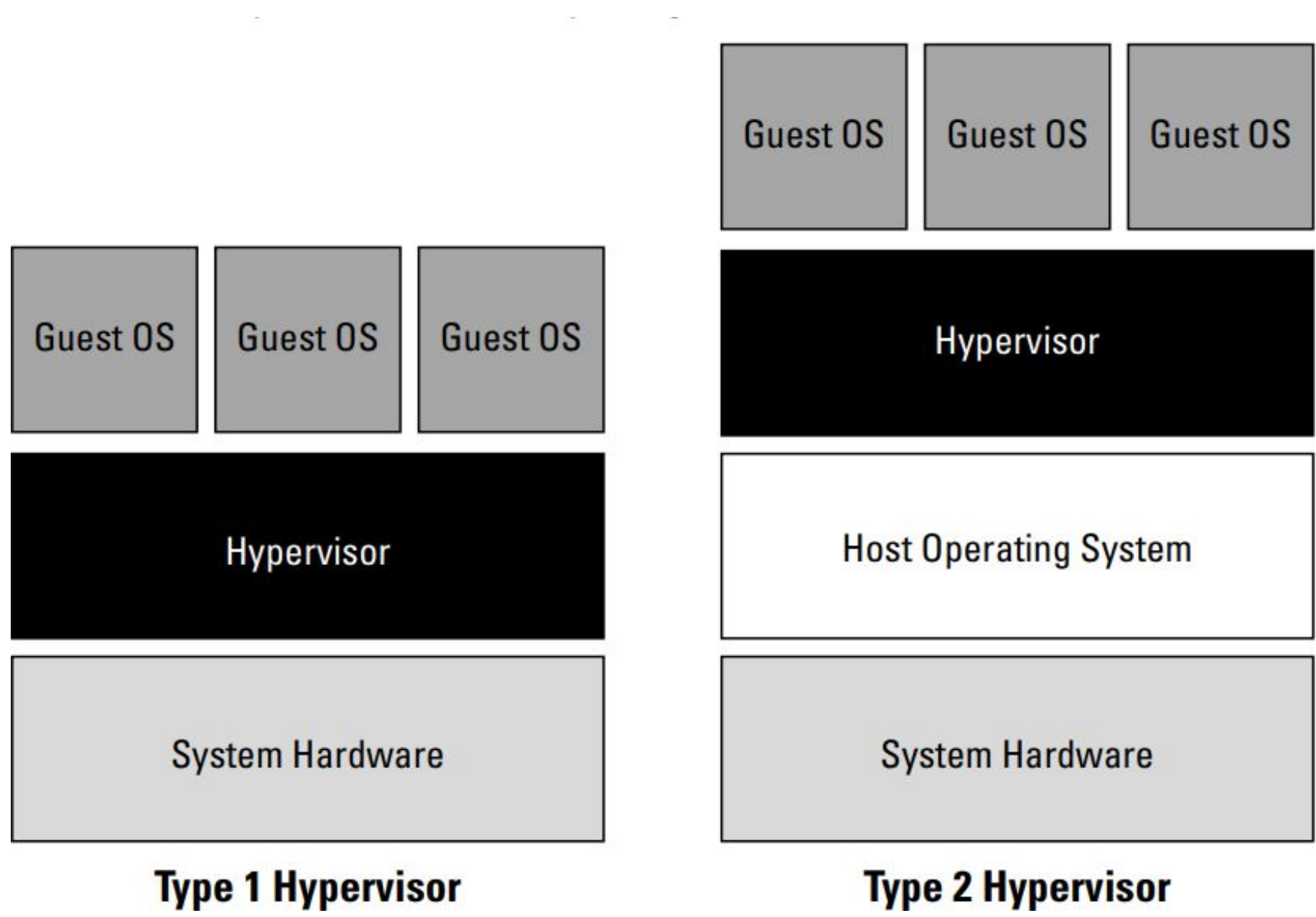
KVM simpler and smaller than hypervisors that take control of the entire machine.

KVM leverages **hardware-assisted virtualization**, which improves performance and allows it to support unmodified guest operating systems.

Currently, KVM supports several versions of Windows, Linux, and UNIX.

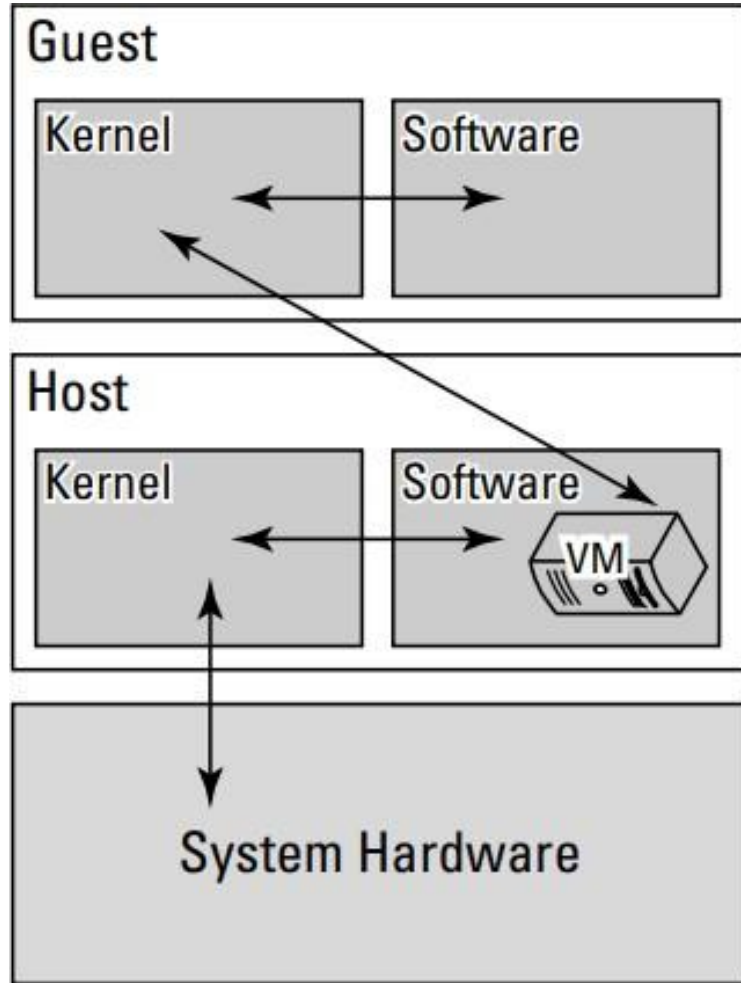


# Cloud Computing Virtualization Types





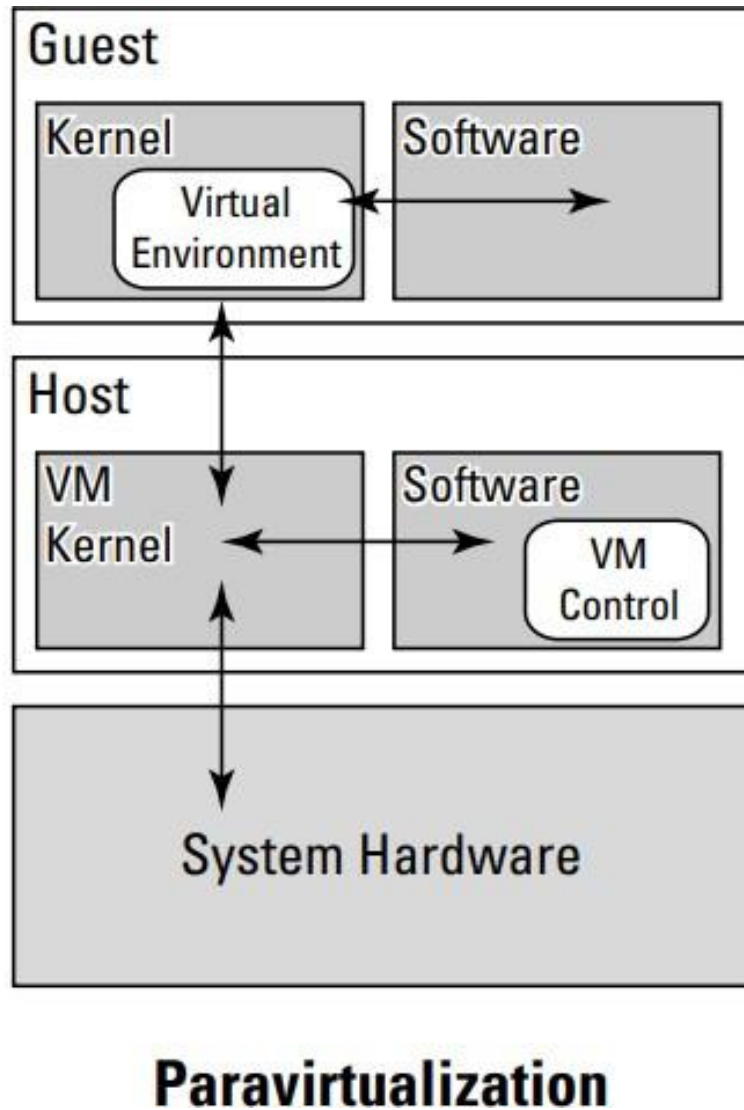
# Cloud Computing Virtualization Types



**Emulation**

In **emulation**, the virtual machine simulates hardware, so it can be independent of the underlying system hardware. A guest operating system using emulation does not need to be modified in any way.

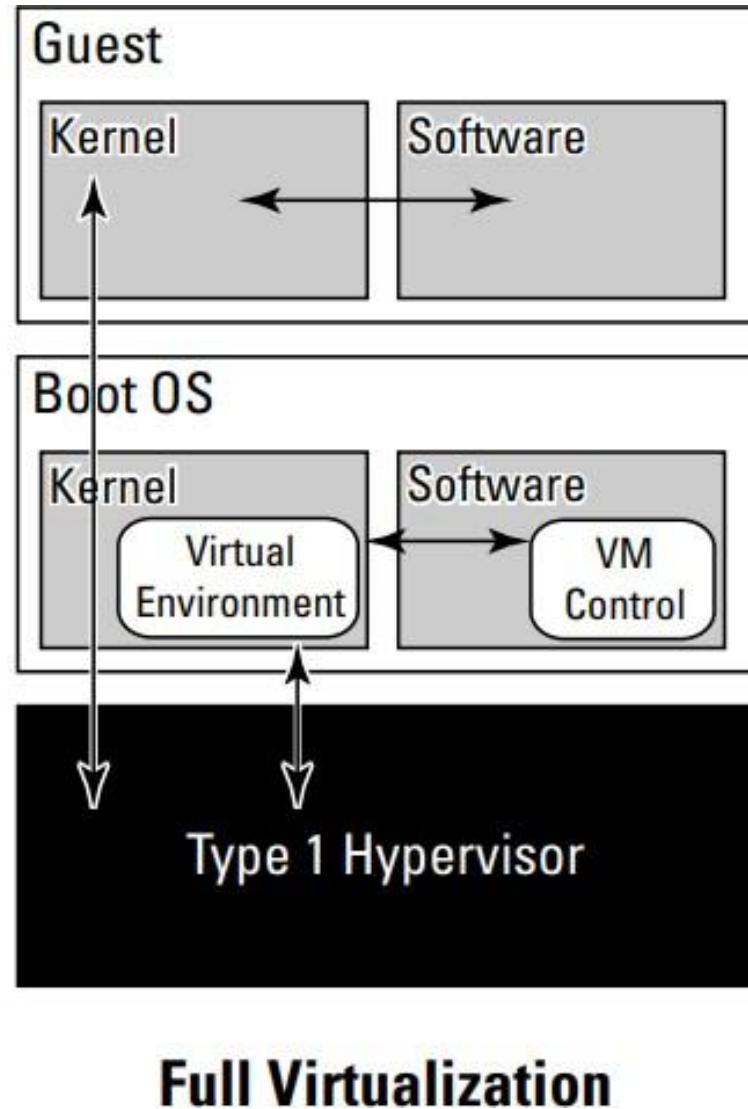
# Cloud Computing Virtualization Types



**Paravirtualization** requires that the host operating system provide a virtual machine interface for the guest operating system and that the guest access hardware through that host VM.

An operating system running as a guest on a paravirtualization system must be ported to work with the host interface.

# Cloud Computing Virtualization Types



In a **full virtualization** scheme, the VM is installed as a Type 1 Hypervisor directly onto the hardware.

All operating systems in full virtualization communicate directly with the VM hypervisor, so guest operating systems do not require any modification.

Virtualization	Emulation
The virtual machine may execute the code directly that's available in various languages.	An emulator needs an interpreter to translate the source code.
In virtualization, hardware may be directly accessible.	In terms of emulation, we need a software connector to access hardware.
Virtual Machines solutions are costlier than the emulator.	It is comparatively cheaper than virtualization.
Virtual Machines are relatively quicker in their operation.	Emulators are comparatively slower than virtualization.
Virtualization offers better backup features.	Emulation falls short of virtualization as far as backup and recovery are considered.

## **EMULATION**

In computing, the emulator is a hardware or software that enables one device (named Host) to function like other systems (named Guest) It is a perfect way to execute the hardware and software in any system Emulation brings greater overhead, but it also has its benefits It is relatively inexpensive, easily accessible and allows us to run programs that have become redundant in the available system

## **VIRTUALIZATION**

It is developing a virtual instance of computing resources, including a computer, server, or other hardware components, or a software based resource, such as an operating system A single physical system is divided into many "virtual" servers by virtualization Virtual machines (VMs) run on dedicated hardware without relying on each other We split a single physical device into separate independent worlds, known as virtual machines, through virtualization It allows us to create several computer simulations. from the host hardware with dedicated resources

# Cloud Computing Virtualization Characteristics & Types

## Characteristics of Virtualization

- ❑ **Increased Security:** The ability to control the execution of a guest program in a completely transparent manner opens new possibilities for delivering a secure, controlled execution environment. All the operations of the guest programs are generally performed against the virtual machine, which then translates and applies them to the host programs.
- ❑ **Managed Execution:** In particular, sharing, aggregation, emulation, and isolation are the most relevant features.
- ❑ **Sharing:** Virtualization allows the creation of a separate computing environment within the same host.
- ❑ **Aggregation:** It is possible to share physical resources among several guests, but virtualization also allows aggregation, which is the opposite process. Virtualization for aggregation combines physical servers and their memory and CPU power to create a single, large virtual machine.

## Types of Virtualization

- ✓ Application Virtualization
- ✓ [Network Virtualization](#)
- ✓ Desktop Virtualization
- ✓ Storage Virtualization
- ✓ [Server Virtualization](#)
- ✓ Data virtualization

# Cloud Computing Virtualization Types [What is Virtualization? | IBM](#)

## 1. Application Virtualization:

Application virtualization helps a user to have remote access to an application from a server. The server stores all personal information and other characteristics of the application but can still run on a local workstation through the internet. An example of this would be a user who needs to run two different versions of the same software. Technologies that use application virtualization are hosted applications and packaged applications.

[What is Application Virtualization? ⚙️ Example, Types, Benefits \(wallarm.com\)](#)

## 2. Network Virtualization:

The ability to run multiple virtual networks with each having a separate control and data plan. It co-exists together on top of one physical network. It can be managed by individual parties that are potentially confidential to each other. Network virtualization provides a facility to create and provision virtual networks, logical switches, routers, [firewalls](#), load balancers, [Virtual Private Networks \(VPN\)](#), and workload security within days or even weeks.

[What is Network Virtualization? | VMware Glossary | IN](#)

## 3. Desktop Virtualization:

Desktop virtualization allows the users' OS to be remotely stored on a server in the data center. It allows the user to access their desktop virtually, from any location by a different machine. Users who want specific operating systems other than Windows Server will need to have a virtual desktop. The main benefits of desktop virtualization are user mobility, portability, and easy management of software installation, updates, and patches.

[What is Desktop Virtualization? | IBM](#)

# Cloud Computing Virtualization Types

## **.4. Storage Virtualization:**

Storage virtualization is an array of servers that are managed by a virtual storage system. The servers aren't aware of exactly where their data is stored and instead function more like worker bees in a hive. It makes managing storage from multiple sources be managed and utilized as a single repository. Storage virtualization software maintains smooth operations, consistent performance, and a continuous suite of advanced functions despite changes, breaks down, and differences in the underlying equipment. Your data is written to a virtual drive rather than a real hard drive.

[Storage Virtualization in Cloud Computing – How it Works \(Use Cases\) \(cloudinfrastructureservices.co.uk\)](https://cloudinfrastructureservices.co.uk)

## **5. Server Virtualization:**

This is a kind of virtualization in which the masking of server resources takes place. Here, the central server (physical server) is divided into multiple different virtual servers by changing the identity number, and processors. So, each system can operate its operating systems in an isolated manner. Where each sub-server knows the identity of the central server. It causes an increase in performance and reduces the operating cost by the deployment of main server resources into a sub-server resource. It's beneficial in virtual migration, reducing energy consumption, reducing infrastructural costs, etc. [What is Server Virtualization? The Ultimate Guide | Liquid Web](#)

**6.Data Virtualization:** This is the kind of virtualization in which the data is collected from various sources and managed at a single place without knowing more about the technical information like how data is collected, stored & formatted then arranged that data logically so that its virtual view can be accessed by its interested people and stakeholders, and users through the various cloud services remotely. Many big giant companies are providing their services like Oracle, IBM, At scale, Cdata, etc. [13 Examples Of Useful Data Virtualisation Tools | Indeed.com India](#) [Data Virtualization: 8 Industry-specific Use Cases in 2023 \(aimultiple.com\)](#)



# Cloud Computing XML Basics

XML based web services interfaces are the primary way that the cloud connects containers

XML stands for eXtensible Markup Language

Two parts of XML are:

- ✓ A set of instructions that you add to a collection of words, pictures, and so on, that controls their on screen appearance, formatting, and behavior
- ✓ Tags that you define and embed in the content, and then write programs that agree on how data is defined within the context of your container **Tags are case sensitive.**

If many different containers or services all use the same language to explain to each other what they do and how they can be used, these services can much more easily talk, connect, and send messages to each other

- A **mashup** is an application or Web page that combines data from two or more sources. Ajax (Asynchronous JavaScript and XML) is often used to create mashups.
- The most commonly used set of **protocols** uses XML as the messaging format, the Simple Object Access Protocol (SOAP) protocol as the object model, and a set of discovery and description protocols based on the Web Services Description Language (WSDL) to manage transactions.

# Cloud Computing XML Basics

Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable.

The design goals of XML focus on simplicity, generality, and usability across the Internet. It is a textual data format with strong support via Unicode for different human languages.

- XML stands for extensible Markup Language
- XML is a markup language like HTML
- XML is designed to store and transport data
- XML is designed to be self-descriptive

# Cloud Computing XML Basics

## XML Vs HTML

XML and HTML were designed with different goals:

- XML is designed to carry data emphasizing on what type of data it is.
- HTML is designed to display data emphasizing on how data looks
- XML tags are not predefined like HTML tags.
- HTML is a markup language whereas XML provides a framework for defining markup languages.
- 
- HTML is about displaying data, hence it is static whereas XML is about carrying information, which makes it dynamic.

# Cloud Computing XML Basics

```
<!DOCTYPE html>
<html>
<h1>Note</h1>
  <body>
    <p>To:ABC
    <br>
    From:XYZ
    </p>
    <h2>Meeting</h2>
    <p>Monday 4pm</p>
  </body>
</html>
```

```
<note>
  <to>ABC</to>
  <from>XYZ</from>
  <heading>Meeting</heading>
  <body> Monday 4pm</body>
</note>
```

# Cloud Computing XML Basics

## XML Does Not Use Predefined Tags

The tags in the example above (like `<to>` and `<from>`) are not defined in any XML standard.

These tags are "invented" by the author of the XML document.

HTML works with predefined tags like `<p>`, `<h1>`, `<table>`, etc.

With XML, the author must define both the tags and the document structure.

## XML is Extensible

Most XML applications will work as expected even if new data is added (or removed).

Imagine an application designed to display the original version of note.xml (`<to>` `<from>` `<heading>` `<body>`).

Then imagine a newer version of note.xml with added `<date>` and `<hour>` elements, and a removed `<heading>`.

The way XML is constructed, older version of the application can still work:

```
<note>  
  <date>2023-07-01</date>  
  <hour>08:30</hour>  
  <to>ABC</to>  
  <from>XYZ</from>  
  <body>Monday at  
    4pm</body>  
</note>
```

# Cloud Computing Web Services

- *Web-based apps are developed using a range of programming platforms in today's corporate world.*
- *Some applications are written in Java, others in .Net, and still others in Angular JS, Node.js, and other frameworks. Most of the time, these diverse programs require some form of communication to work together.*
- *Because they are written in separate programming languages, ensuring accurate communication between them becomes extremely difficult. Web services have a role in this.*
- *Web services provide a common platform, for several applications written in different programming languages, to connect with one another*

Customers/clients access cloud services. Server application needs to respond to client application.

Web service is a used to exchange messages between client and server applications.

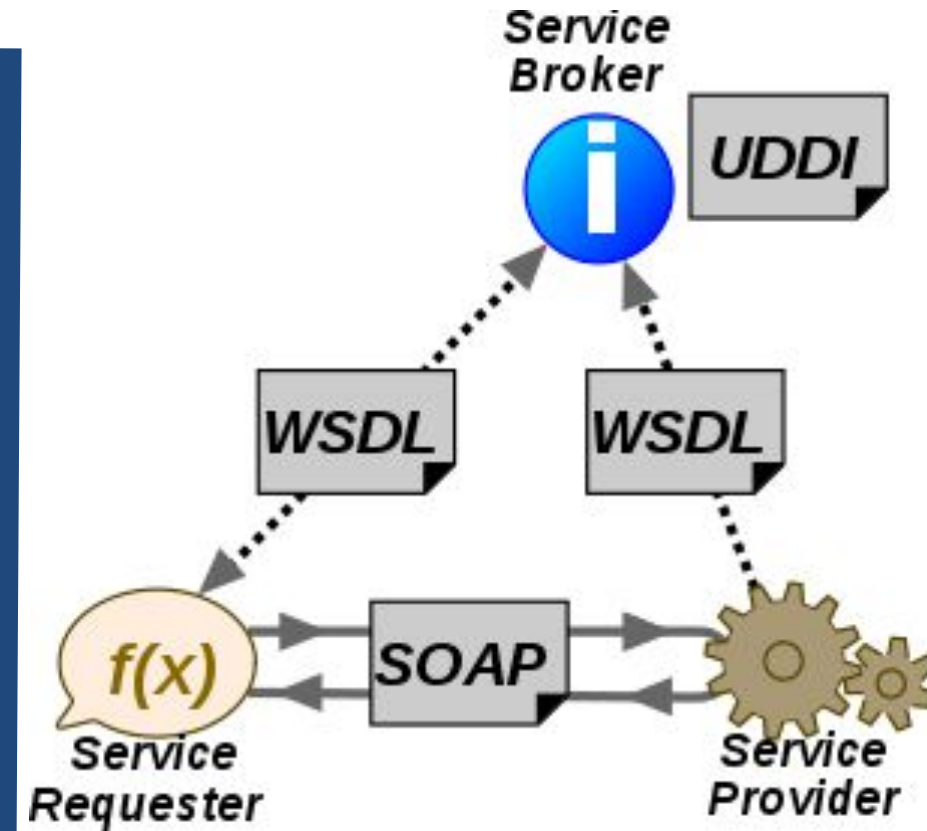
# Cloud Computing Web Services

- A web service is a **set of open protocols** and standards that allow data to be exchanged between different applications or systems.
- Web services can be used by software programs written in a variety of programming languages and running on a variety of platforms to exchange data via computer networks such as the Internet in a similar way to inter-process communication on a single computer.
- Any software, application, or cloud technology that uses standardized web protocols (HTTP or HTTPS) to connect, interoperate, and exchange data messages – commonly XML (Extensible Markup Language) – across the internet is considered a web service.
- Web services have the advantage of allowing programs developed in different languages to connect with one another by exchanging data over a web service between clients and servers.

A client invokes a web service by submitting an XML request, which the service responds with an XML response.

# Web Services Architecture

- A web service is a software system designed to support interoperable machine-to-machine interaction over a network.
- It has an interface described in a machine-processable format (specifically [WSDL](#)).  
Web Services Description Language
- Other systems interact with the web service in a manner prescribed by its description using SOAP(Simple Object Access Protocol)-messages, typically conveyed using [HTTP](#) with an [XML serialization](#) in conjunction with other web-related standards.
- UDDI (Universal Description, Discovery, and Integration)



**Web services architecture**



# Web Services Architecture

A web service is a software system designed to support interoperable machine-to-machine interaction over a network.

It has an interface described in a machine-processable format (specifically [WSDL](#)).

Web Services Description Language

Other systems interact with the web service in a manner prescribed by its description using SOAP(Simple Object Access Protocol)-messages, typically conveyed using [HTTP](#) with an [XML serialization](#) in conjunction with other web-related standards.

UDDI (Universal Description, Discovery, and Integration)

## Web services architecture:

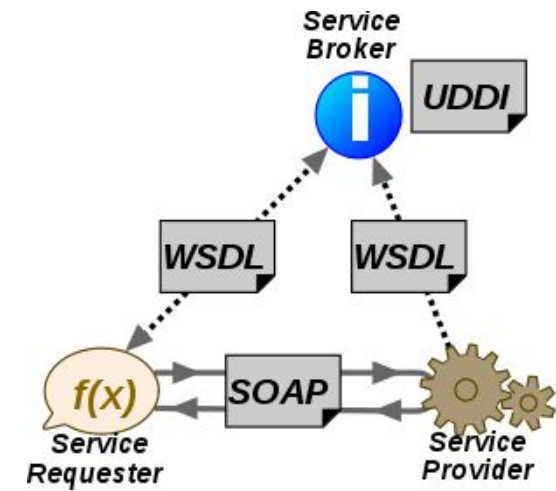
The service provider sends a WSDL file to UDDI.

The service requester contacts UDDI to find out who is the provider for the data it needs, and then it contacts the service provider using the SOAP protocol.

The service provider validates the service request and sends structured data in an XML file, using the SOAP protocol.

This XML file would be validated again by the service requester using an XSD file.

**NOTE:** An XML schema definition (XSD), is a framework document that defines the rules and constraints for XML documents. An XSD formally describes the elements in an XML document and can be used to validate the contents of the XML document to make sure that it adheres to the rules of the XSD.



**Web services architecture**

# Web Services: UDDI

UDDI (Universal Description, Discovery, and Integration)

UDDI is a standard for specifying, publishing and discovering a service provider's online services.

The UDDI registry will hold the required information for the online service, just like a telephone directory has the name, address, and phone number of a certain individual.

So that a client application may figure out where it is.

It provides a specification that aids in the hosting of data via web services.

UDDI provides a repository where WSDL files can be hosted so that a client application can discover a WSDL file to learn about the various actions that a web service offers.

As a result, the client application will have full access to the UDDI, which **serves as a database for all WSDL files.**

# Web Services:

## WSDL

### WSDL (Web Services Description Language)

- If a web service can't be found, it can't be used.
- The client invoking the web service should be aware of the location of the web service.
- Second, the client application must understand what the web service does in order to invoke the correct web service.
- The WSDL, or Web services description language, is used to accomplish this.
- The WSDL file is another XML-based file that explains what the web service does to the client application.
- The client application will be able to understand where the web service is located and how to use it by using the WSDL document.

# Web Services:

## SOAP

### SOAP (Simple Object Access Protocol)

- It is a transport-independent **messaging protocol**.
- SOAP is built on sending XML data in the form of SOAP Messages.
  - A document known as an XML document is attached to each message.
- The best thing about Web services and SOAP is that everything is sent through HTTP, the standard web protocol.
- A root element known as the element is required in every SOAP document.
- In an XML document, the root element is the first element. The header comes first, followed by the body.
- The routing data, or information that directs the XML document to which client it should be sent to, is contained in the **header**.
- The real message will be in the **body**.
-

# Cloud Computing Service Oriented Architecture

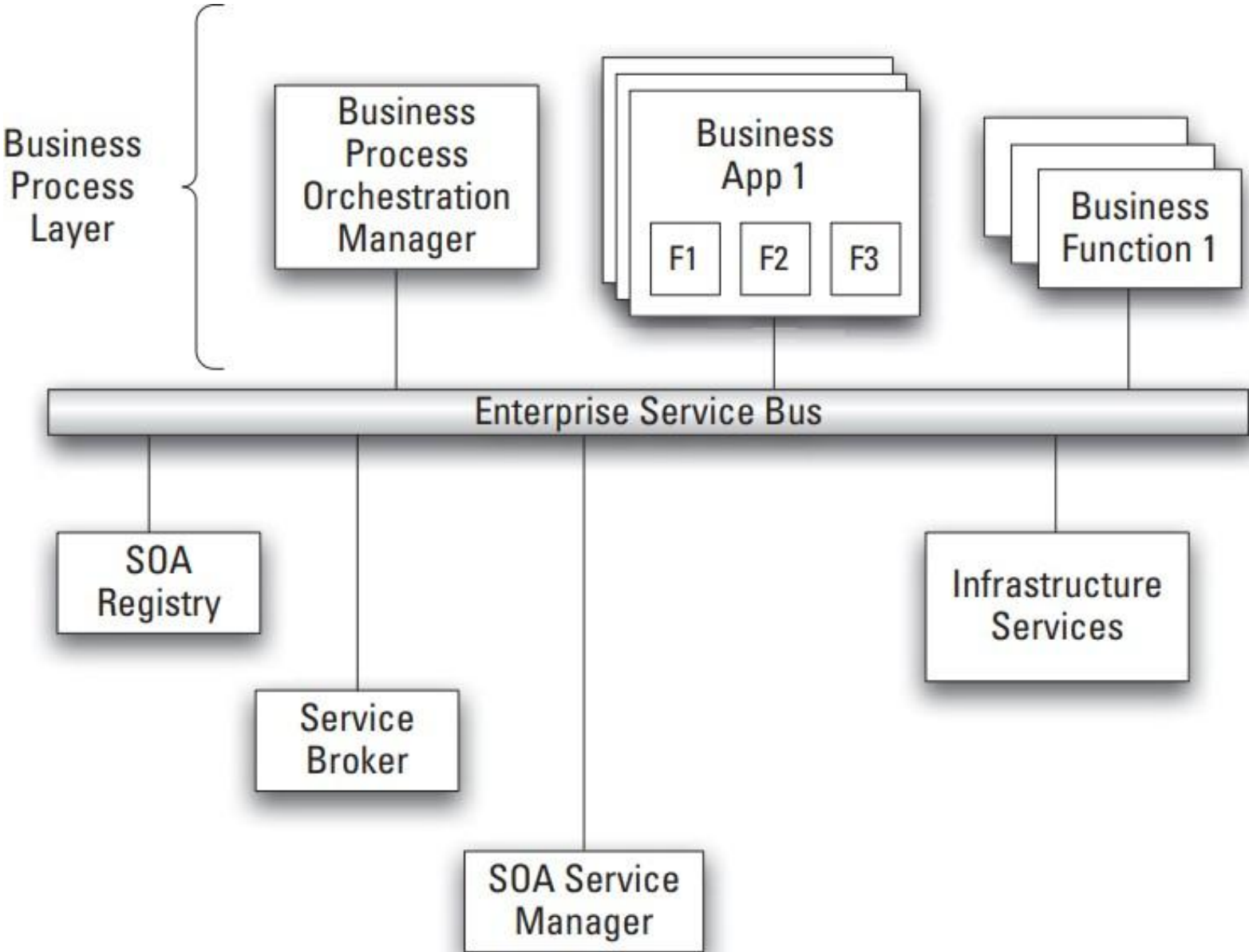
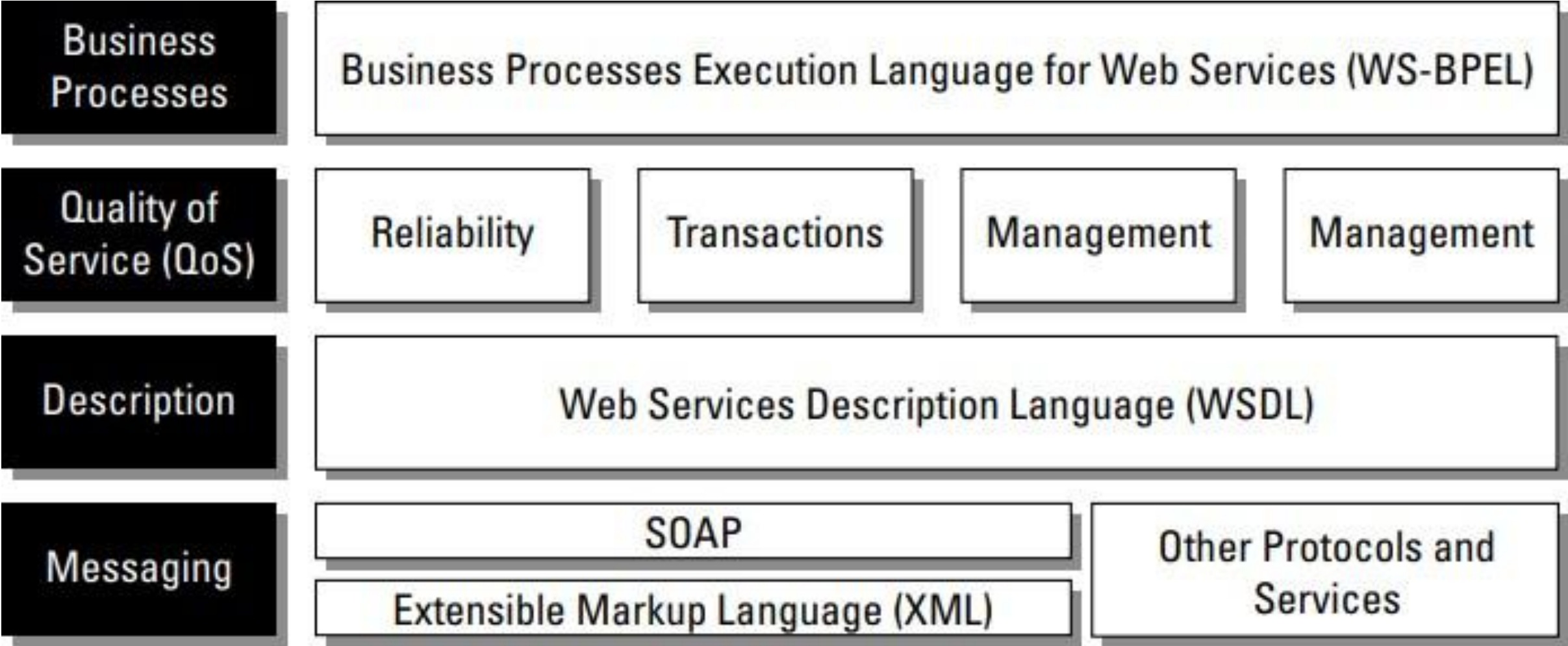


Figure: Fundamentals of SOA components

# Cloud Computing Service Oriented Architecture

- The **Enterprise Service Bus** – **ESB** is a collection of software components that manage messaging from one software component to another.
- A software component connects to the ESB and passes it a message by using a specified format along with the address of the software component that needs to receive the message.
- The ESB completes the job, getting the message from the sending component to the receiving component.

# Service Oriented Architecture: Protocols



# Service Oriented Architecture: Protocols

Figure: A protocol stack for SOA showing the relationship of each protocol to its function

- SOA requires the use of an orchestrator or broker service to ensure that messages are correctly transacted.
- In the figure, the box labeled Other Services could include
  - Common Object Request Broker Architecture (CORBA),
  - Representational State Transfer (REST),
  - Remote Procedure Calls (RPC),
  - Distributed Common Object Model (DCOM), and other technologies and protocols.



# Cloud Computing Service Oriented Architecture

Service Oriented Architecture (SOA) describes **a standard method for requesting services** from distributed components and managing the results

Clients requesting services, the components providing the services, the protocols used to deliver messages, and the responses vary widely

So **SOA provides the translation and management layer** in an architecture that removes the barrier for a client obtaining desired services

With SOA, clients and components can be written in different languages and can use multiple messaging protocols and networking protocols to communicate with one another

**SOA provides the standards** that transport the messages and makes the infrastructure to support it

SOA is a **way of designing software** to leverage the possibilities of programming in individual, independent yet loosely coupled services.

SOA provides access to reusable Web services over a TCP/IP network, which makes this an important topic to cloud computing going forward

# Service Oriented Architecture: When to skip?

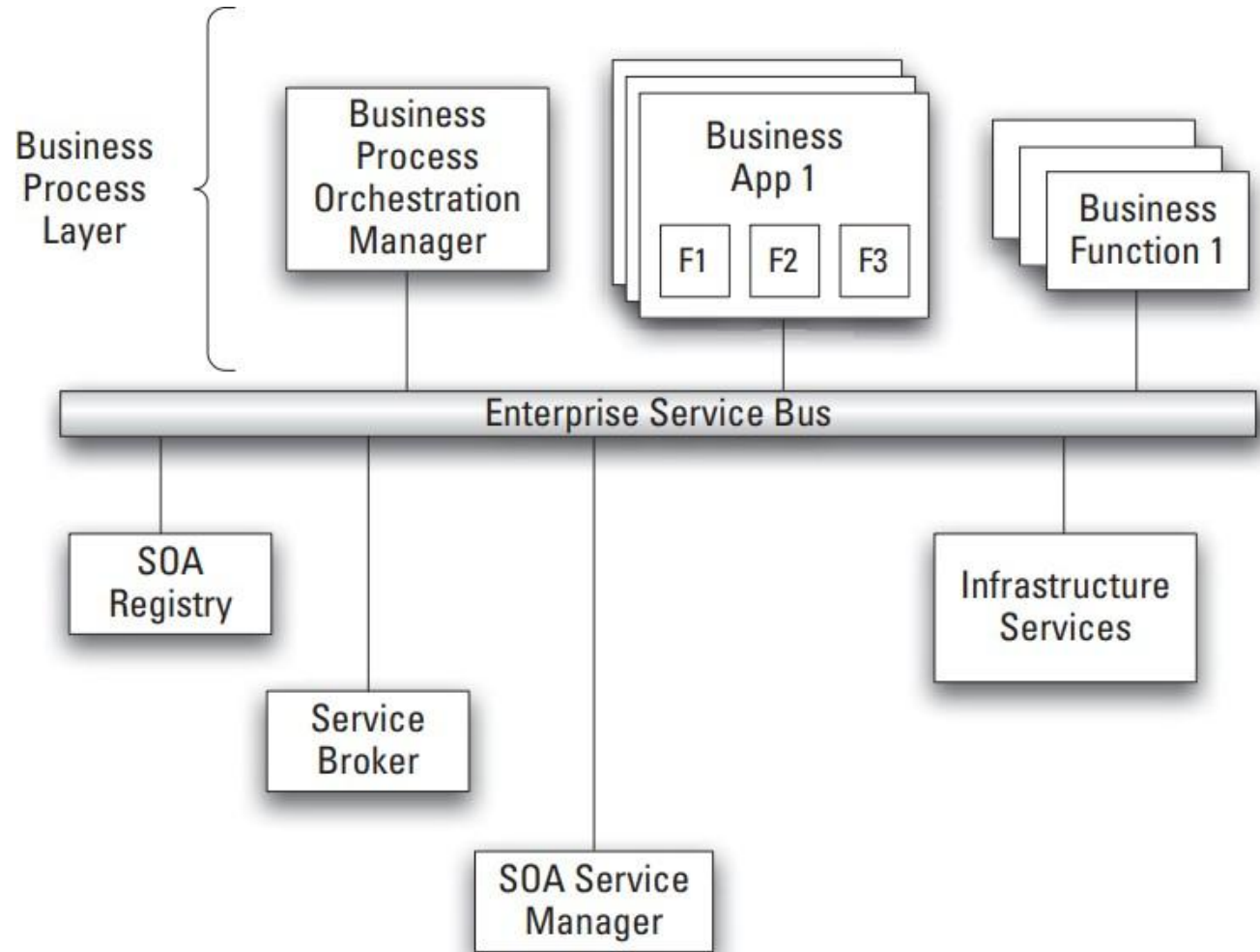
- You don't need SOA if you are creating a monolithic cloud application.
- Monolithic application is a **single unified software application which is self contained and independent** from other applications, but typically lacks flexibility.
- Monolithic application performs specific function such as backup, e mail, Web page access, or instant messaging.
- Many of the large and familiar cloud computing applications are monolithic and were built with proprietary technologies albeit often on top of open source software and hardware.
- However, as cloud computing applications expand their capability to provide additional and diverse services, SOA offers access to ready made, modular, highly optimized, and widely shareable components that can minimize developer and infrastructure costs

CP/M and DOS are simple examples of monolithic operating systems. Both CP/M and DOS are operating systems that share a single address space with the applications.

Amazon Prime, a widely-used streaming platform, recently transitioned from a microservices architecture to a monolithic one. This shift was driven by the need for improved efficiency and reduced operational complexity.

Most of the successful startups we read about (Netflix, Uber, Airbnb, etc.), all started with a monolith architecture. Once they became ready for scale, they switched to microservices.

# Service Oriented Architecture: SOA Registry



## SOA Registry

Information describing the function of each reusable component is recorded in the SOA registry – a type of electronic catalog.

The SOA registry has two roles:

- ✓ One rooted in the operational environment
- ✓ One rooted in the world of programmers and business analysts

Figure: Fundamentals of SOA components

The SOA registry contains information (metadata) about all the components that the SOA supports.

# Service Oriented Architecture: SOA Registry

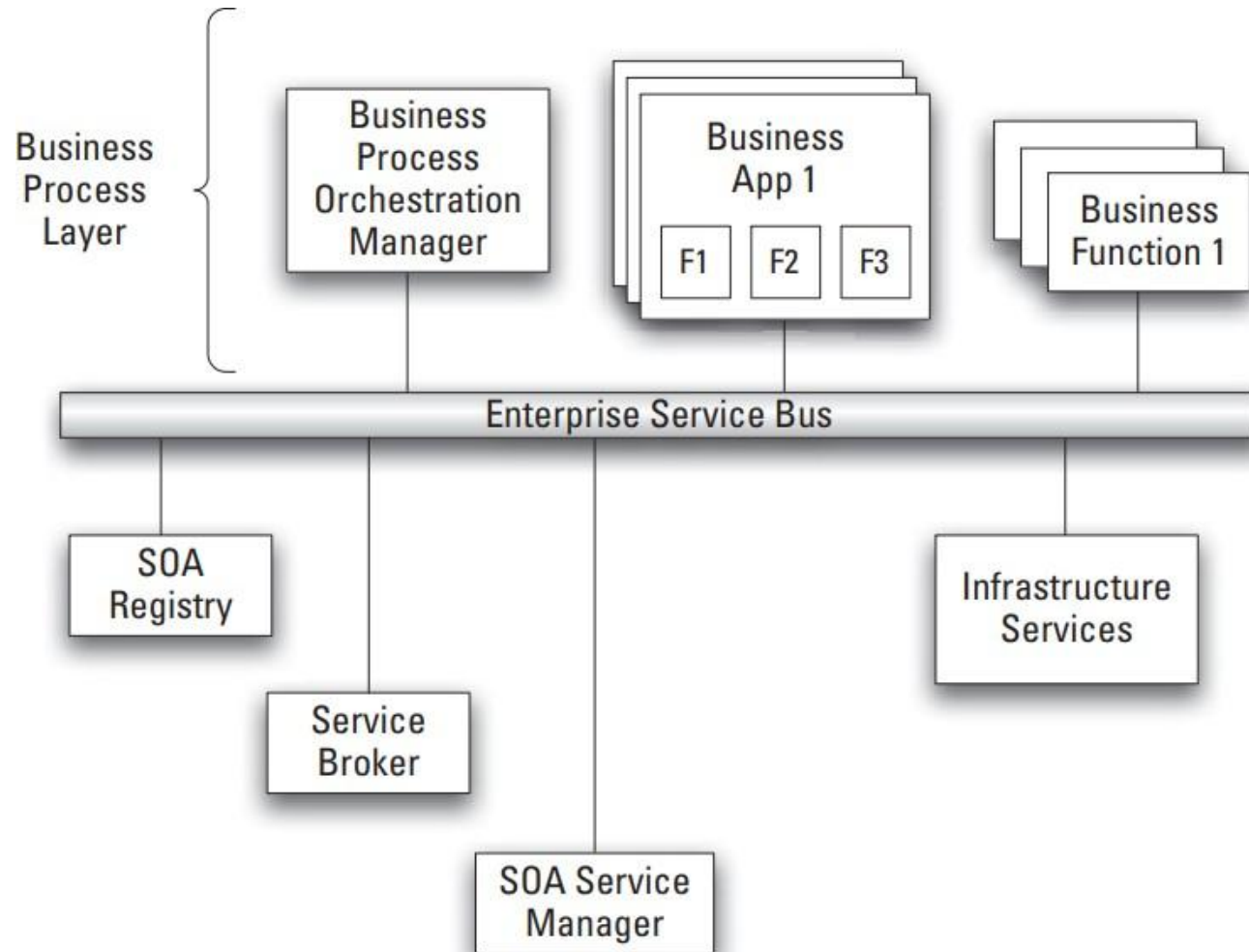


Figure: Fundamentals of SOA components

## SOA Registry

### ✓ One rooted in the operational environment:

In the day-to-day working business computing environment, the SOA registry provides reference information about software components that are running or available for use. This information is of particular importance to the **service broker** – the software in a SOA framework that brings components together by using the rules associated with each component.

### ✓ One rooted in the world of programmers and business analysts:

For programmers and business analysts, on the other hand, the SOA registry acts as a reference that helps them select components and then connect them to create composite applications that represent business processes. It also stores information about how each component connects to other components. In other words, the SOA registry documents the rules and descriptions associated with every given component.

# Characterizing SOA

The principal characteristics of SOA are described in more detail here:

- ✓ SOA is a black-box component architecture.
- ✓ SOA components are loosely coupled.
- ✓ SOA components are orchestrated to link through business processes to deliver a well-defined level of service.

# Characterizing SOA

✓ SOA is a black box component architecture

The black box lets you reuse existing business applications;

It simply adds a fairly simple adapter to them

You don't need to know every detail of what's inside each component;

SOA hides the complexity whenever possible

# Characterizing SOA

## ✓ SOA components are loosely coupled

Software components are loosely coupled if they're designed to interact in a standardized way that minimizes dependencies

One loosely coupled component passes data to another component and makes a request; the second component carries out the request and, if necessary, passes data back to the first

Each component offers a small range of simple services to other components

A set of loosely coupled components does the same work that software components in tightly structured applications used to do, but with loose coupling you can combine and recombine the components in a bunch of ways

This makes a world of difference in the ability to make changes easily, accurately, and quickly

# Characterizing SOA

✓ SOA components are orchestrated to link through business processes to deliver a well defined level of service

SOA creates a simple arrangement of components that, together, deliver a very complex business service

SOA must provide acceptable service levels

To that end, the components ensure a dependable service level

Service level is tied directly to the best practices of conducting business, commonly referred to as business process management (BPM)

BPM focuses on effective design of business process and SOA allows IT to align with business processes