Chapter-2

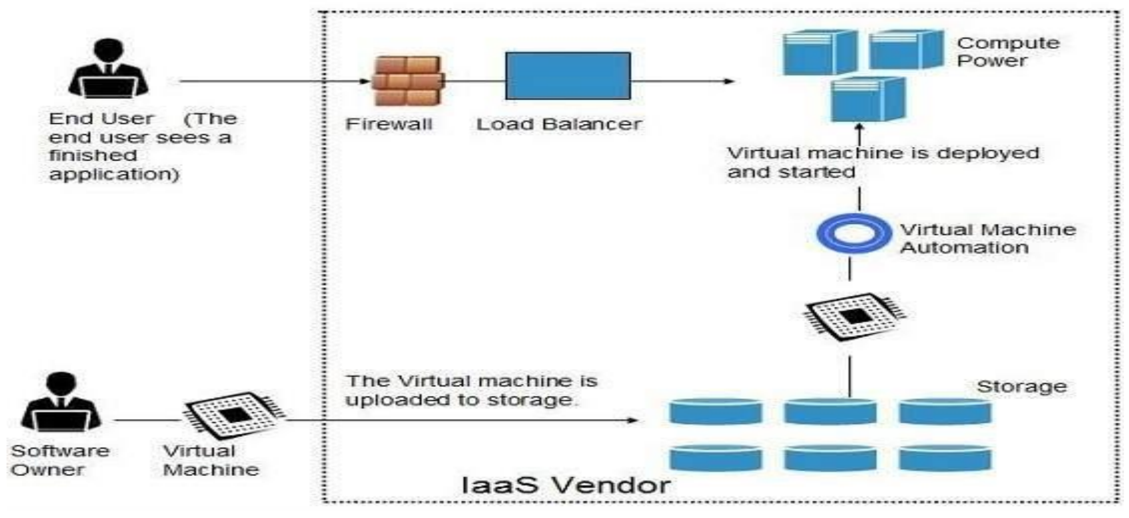
Cloud Service Models

2.1Infrastructure-as-a-Service

It provides access to fundamental resources such as physical machines, virtual machines, virtual storage, etc. Apart from these resources, the IaaS also offers:

- Virtual machine disk storage
- Virtual local area network (VLANs)
- Load balancers
- IP addresses
- Software bundles

All of the above resources are made available to end user via **server virtualization.** Moreover, these resources are accessed by the customers as if they own them.


End User (The end user sees a finished application)
Firewall
Load Balancer
Compute Power
Virtual machine is deployed and started
Virtual Machine Automation
The Virtual machine is uploaded to storage.
Storage
Software Owner
Virtual Machine
IaaS Vendor

Advantage of IaaS:

IaaS allows the cloud provider to freely locate the infrastructure over the Internet in a cost-effective manner. Some of the key benefits of IaaS are listed below:

- Full control of the computing resources through administrative access to VMs.
- Flexible and efficient renting of computer hardware.
- Portability, interoperability with legacy applications.

Full control over computing resources through administrative access to VMs:

IaaS allows the customer to access computing resources through administrative access to virtual machines in the following manner:

- Customer issues administrative command to cloud provider to run the virtual machine or to save data on cloud server.
- Customer issues administrative command to virtual machines they owned to start web server or to install new applications.

Flexible and Efficient Renting of Computer Hardware:

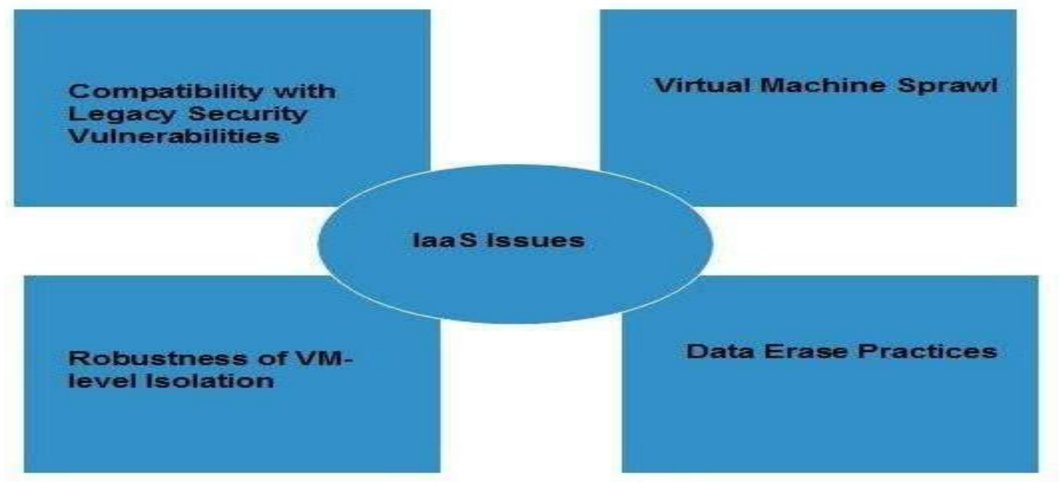
IaaS resources such as virtual machines, storage devices, bandwidth, IP addresses, monitoring services, firewalls, etc. are made available to the customers on rent. The payment is based upon the amount of time the customer retains a resource. Also with administrative access to virtual machines, the customer can run any software, even a custom operating system.

Portability, Interoperability with Legacy Applications:

It is possible to maintain legacy between applications and workloads between IaaS clouds. For example, network applications such as web server or e-mail server that normally runs on customer-owned server hardware can also run from VMs in IaaS cloud.

Issues:

IaaS shares issues with PaaS and SaaS, such as Network dependence and browser based risks. It also has some specific issues, which are mentioned in the following diagram:


Compatibility with Legacy Security Vulnerabilities
Virtual Machine Sprawl
IaaS Issues
Robustness of VM-level Isolation
Data Erase Practices

a. **Compatibility with Legacy Security Vulnerabilities:**

Because IaaS offers the customer to run legacy software in provider's infrastructure, it exposes customers to all of the security vulnerabilities of such legacy software.

b. **Virtual Machine Sprawl:**

The VM can become out-of-date with respect to security updates because IaaS allows the customer to operate the virtual machines in running, suspended and off state. However, the provider can automatically update such VMs, but this mechanism is hard and complex.

c. **Robustness of VM-level Isolation:**

IaaS offers an isolated environment to individual customers through hypervisor. Hypervisor is a software layer that includes hardware support for virtualization to split a physical computer into multiple virtual machines.

d. **Data erase practices**

The customer uses virtual machines that in turn use the common disk resources provided by the cloud provider. When the customer releases the resource, the cloud provider must ensure that next customer to rent the resource does not observe data residue from previous customer.

Characteristics:

Here are the characteristics of IaaS service model:

- Virtual machines with pre-installed software.
- Virtual machines with pre-installed operating systems such as Windows, Linux, and Solaris.
- On-demand availability of resources.
- Allows to store copies of particular data at different locations.

- The computing resources can be easily scaled up and down.

On-demand Computing:

On-demand computing is a delivery model in which computing resources are made available to the user as needed. The resources may be maintained within the user's enterprise, or made available by a cloud service provider. When the services are provided by a third-party, the term cloud computing is often used as a synonym for on-demand computing.

Amazon's Elastic Cloud:

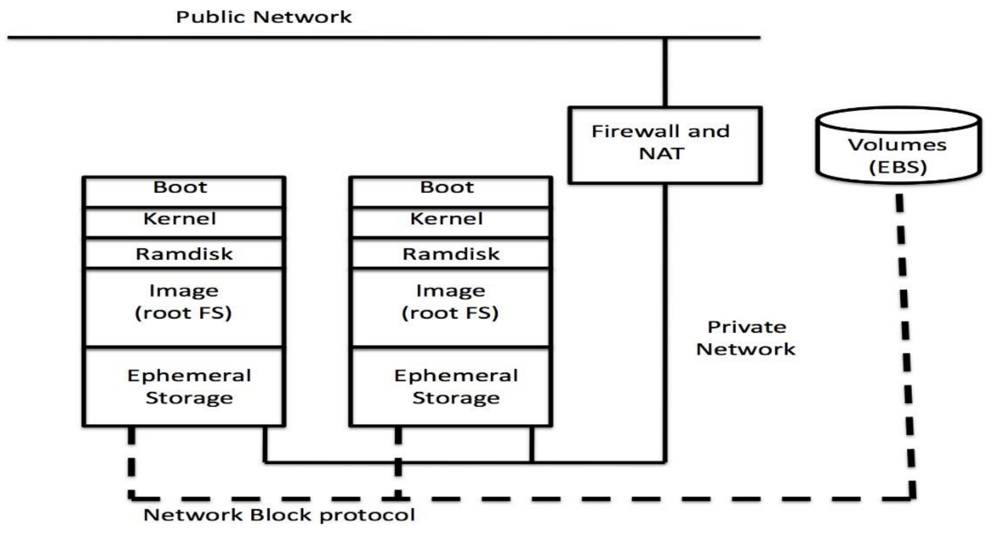
Amazon Elastic Compute Cloud (Amazon EC2) is an Amazon Web Service (AWS) you can use to access servers, software, and storage resources across the Internet in a self-service manner. It provides scalable, pay as-you-go compute capacity. It is said to be an elastic since it is scalable in both direction along the client as well as service provider. Amazon EC2 provides the following features:

- Virtual computing environments, known as *instances*
- Preconfigured templates for your instances, known as *Amazon Machine Images (AMIs)*, that package the bits you need for your server (including the operating system and additional software)
- Various configurations of CPU, memory, storage, and networking capacity for your instances, known as *instance types*
- Secure login information for your instances using *key pairs* (AWS stores the public key, and you store the private key in a secure place)
- Storage volumes for temporary data that's deleted when you stop or terminate your instance, known as *instance store volumes*
- Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS), known as *Amazon EBS volumes*
- Multiple physical locations for your resources, such as instances and Amazon EBS volumes, known as *regions* and *Availability Zones*
- A firewall that enables you to specify the protocols, ports, and source IP ranges that can reach your instances using *security groups*
- Static IPv4 addresses for dynamic cloud computing, known as *Elastic IP addresses*
- Metadata, known as *tags*, that you can create and assign to your Amazon EC2 resources
- Virtual networks you can create that are logically isolated from the rest of the AWS cloud, and that you can optionally connect to your own network, known as *virtual private clouds*(VPCs)

Amazon's EC2 provides essentially three functionalities:

- virtual machine provisioning
- network provisioning (including firewalls)
- block storage provisioning (persistent volumes)

It use the following service model:



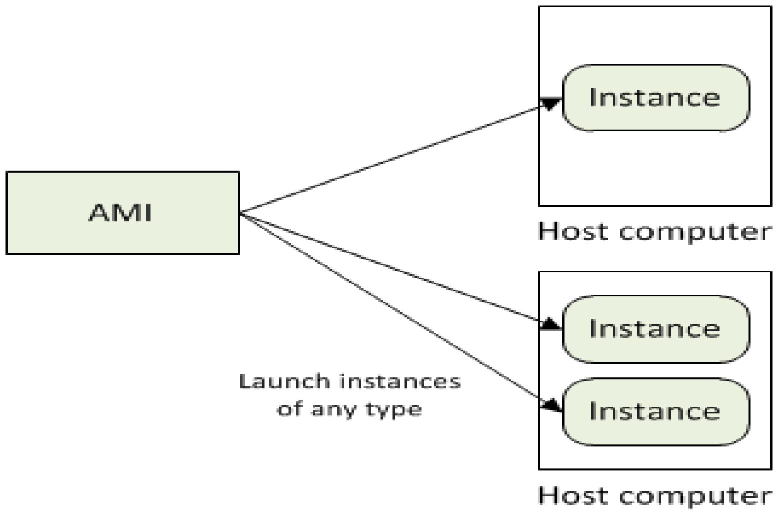
EC2 includes following components:

- AMI & Instance
- Region & Zones
- Storage
- Networking and Security
- Monitoring
- Auto Scaling
- Load Balancer

<u>**AMI and Instance**</u>

Amazon Machine Image (AMI) is a template for software configuration (Operating System, Application Server, and Applications). Amazon publishes many AMI for public use and Custom AMIs provided by community members. In this platform, user can create their own AMI.

Instance is an AMI running on virtual servers in the cloud and instance type specifies the different operating environment. Each *instance type* offers different compute and memory facilities to each user.



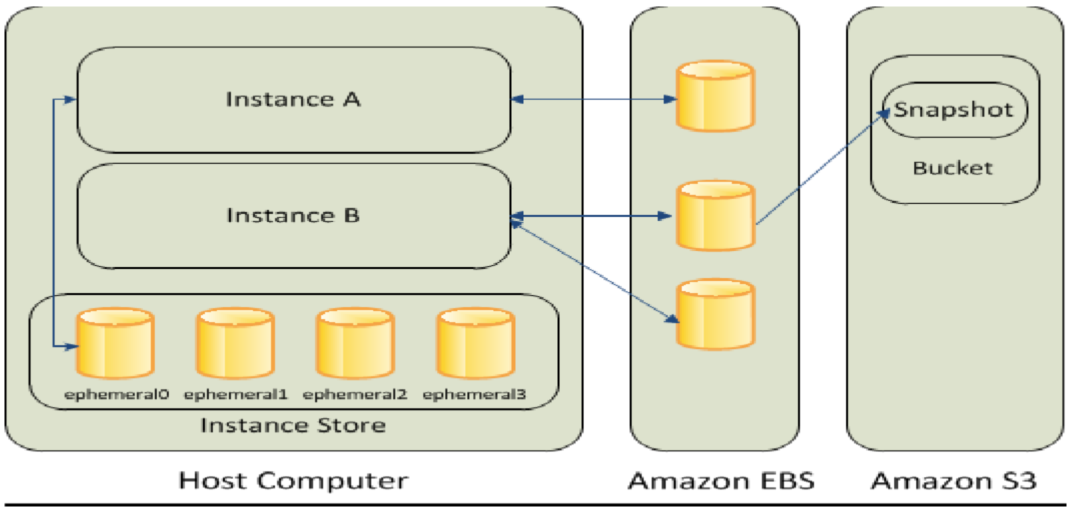
Region and Zones:

Amazon have data centers in different region across the globe for example North America, Europe, and Asia. Prices for Amazon EC2 usage vary by region. An instance can be launched in different regions depending on the need. It depends on

- Closer to specific customer
- To meet legal or other requirements

Each region has set of zones and Zones are isolated from failure in other zones. When each region contains multiple distinct locations then it is called as *Availability Zones*. Inexpensive, low latency connectivity between zones in same region must be maintained to provide the better service.

Storage:



Amazon EC2 provides three type of storage option

- Amazon EBS
- Amazon S3

- Instance Storage

Amazon EBS (Elastic Block Store) provides with persistent, block-level storage. Basically additional Hard Disk that you can attach to instance. It is suitable for apps which require database, file system, and block level storage. User can Create, Attach, Detach, Delete the storage as per requirement.

When the EBS be partitions into no of small dedicated volume then it is called S3 storage. By taking snapshots that is stored in S3, a new EBS can be re-created using the snapshot. S3 simple storage service storage for the Internet or web service interface that enables you to store and retrieve any amount of data from anywhere on the web.

Storage physically attached to the computer is called as Instance Storage. Instance store comes with each instance except the micro-one, temporary block level storage.

Networking and Security:

Instances can be launched on one of the two platforms

- ✓ EC2-Classic
- ✓ EC2-VPC

VPC launch Amazon Web Services (AWS) resources into a virtual network that you've defined. During the Configuration of VPC we select its IP address range, create subnets, and configure route tables, network gateways, and security settings. Instance IP address is dynamic since a new IP address is assigned every time instance is launched.

The Security Group be created that enables you to specify the protocols, ports, and source IP ranges that are allowed to reach your instances. In this model, we can create multiple security groups, assign instance to a particular group, and determine the traffic.

Monitoring, Auto Scaling, and Load Balancing:

In this model. We can monitor statistics of instances and EBS through the Cloud Watch. It is a tools that is used to monitor, manage, and publish various metrics of cloud service.

Amazon EC2 capacity be scaled up and down automatically so that it’s called as elastic and based on following rules

- Add and remove compute resource based on demand
- Suitable for businesses experiencing variability in usage

The no of the load balancers are used to balance incoming traffic. It distributes incoming traffic across multiple instances and corresponding procedure is called Elastic Load Balancing.

<u>Benefits of EC2</u>

1. **Elastic Web-Scale Computing**

Amazon EC2 enables you to increase or decrease capacity within minutes, not hours or days. You can commission one, hundreds or even thousands of server instances simultaneously. Of course, because this is all controlled with web service APIs, your application can automatically scale itself up and down depending on its needs.

2. **Completely Controlled**

You have complete control of your instances. You have root access to each one, and you can interact with them as you would any machine. You can stop your instance while retaining the data on your boot partition and then subsequently restart the same instance using web service APIs. Instances can be rebooted remotely using web service APIs. You also have access to console output of your instances.

3. **Flexible Cloud Hosting Services**

You have the choice of multiple instance types, operating systems, and software packages. Amazon EC2 allows you to select a configuration of memory, CPU, instance storage, and the boot partition size that is optimal for your choice of operating system and application. For example, your choice of operating systems includes numerous Linux distributions, and Microsoft Windows Server.

4. **Designed for use with other Amazon Web Services**

Amazon EC2 works in conjunction with Amazon Simple Storage Service (Amazon S3), Amazon Relational Database Service (Amazon RDS) and Amazon Simple Queue Service (Amazon SQS) to provide a complete solution for computing, query processing and storage across a wide range of applications.

5. **Reliable**

Amazon EC2 offers a highly reliable environment where replacement instances can be rapidly and predictably commissioned. The service runs within Amazon’s proven network infrastructure and datacenters.

6. **Secure**

Amazon EC2 works in conjunction with Amazon VPC to provide security and robust networking functionality for your computer resources. Your compute instances are located in a Virtual Private Cloud (VPC) with an IP range that you specify. You decide which instances are exposed to the Internet and which remain private.

- ➢ Security Groups and networks ACLs allow you to control inbound and outbound network access to and from your instances.
- ➢ You can provision your EC2 resources as Dedicated Instances. Dedicated Instances are Amazon EC2 Instances that run on hardware dedicated to a single customer for additional isolation.
- ➢ If you do not have a default VPC you must create a VPC and launch instances into that VPC to leverage advanced networking features such as private subnets, outbound security group filtering, network ACLs and Dedicated Instances.

7. **Inexpensive**

Amazon EC2 passes on to you the financial benefits of Amazon's scale. You pay a very low rate for the compute capacity you actually consume.

- On-Demand Instances – On-Demand Instances let you pay for compute capacity by the hour with no long-term commitments. This frees you from the costs and complexities of planning, purchasing, and maintaining hardware and transforms what are commonly large fixed costs into much smaller variable costs. On-Demand Instances also remove the need to buy "safety net" capacity to handle periodic traffic spikes.
- Reserved Instances – Reserved Instances provide you with a significant discount compared to On-Demand Instance pricing. There are three Reserved Instance payment options (No Upfront, Partial Upfront, All Upfront) that enable you to balance the amount you pay upfront with your effective hourly price.
- Spot Instances - Spot instances allow you to bid on spare Amazon EC2 computing capacity. Since Spot instances are often available at a discount compared to On-Demand pricing, you can significantly reduce the cost of running your applications, grow your application's compute capacity and throughput for the same budget, and enable new types of cloud computing applications.

8. **Easy to Start**

Quickly get started with Amazon EC2 by visiting the AWS Management Console to choose preconfigured software on Amazon Machine Images (AMIs). You can quickly deploy thissoftware to EC2 via the EC2 console.

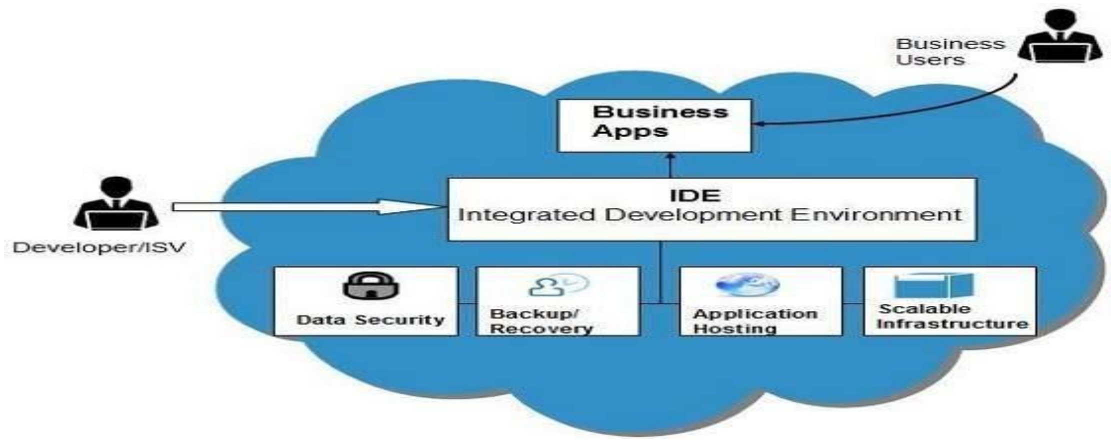
2.2 Platform-as-a-Service :

Platform-as-a-Service offers the runtime environment for applications. It also offers development and deployment tools required to develop applications. PaaS has a feature of point-and-click tools that enables non-developers to create web applications.

App Engine of Google and Force.com are examples of PaaS offering vendors. Developer may log on to these websites and use the built-in API to create web-based applications.

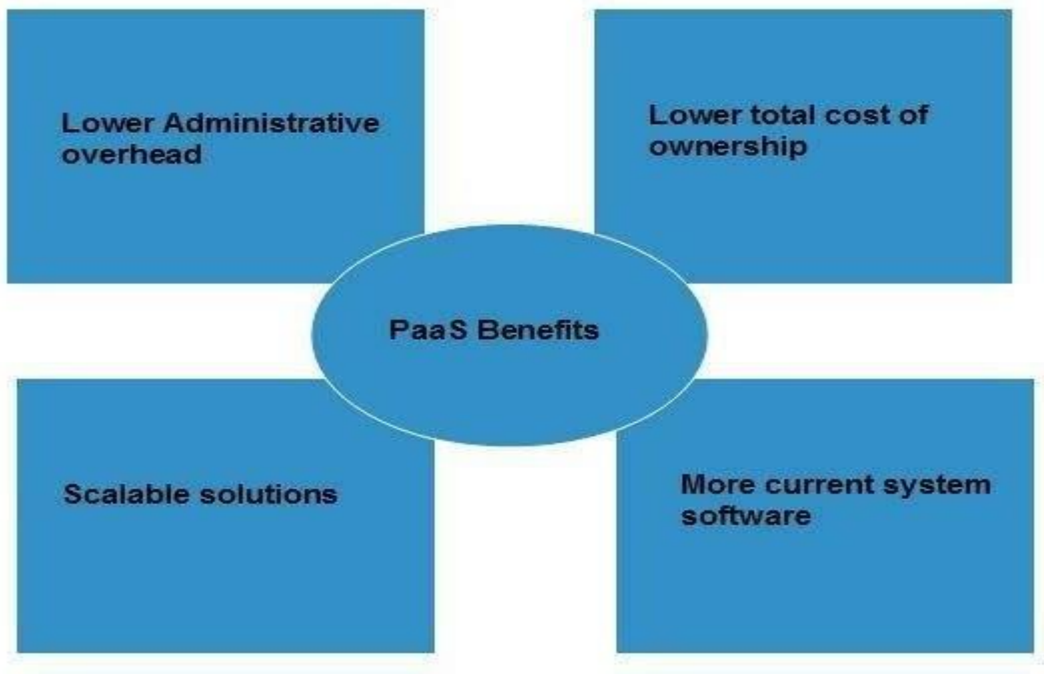
But the disadvantage of using PaaS is that, the developer locks-in with a particular vendor. For example, an application written in Python against API of Google, and using App Engine of Google is likely to work only in that environment.

The following diagram shows how PaaS offers an API and development tools to the developers and how it helps the end user to access business applications.



Benefits:

Following are the benefits of PaaS model:


Lower Administrative overhead
Lower total cost of ownership
PaaS Benefits
Scalable solutions
More current system software

1. Lower administrative overhead

Customer need not bother about the administration because it is the responsibility of cloud provider.

2. Lower total cost of ownership

Customer need not purchase expensive hardware, servers, power, and data storage.

3. Scalable solutions

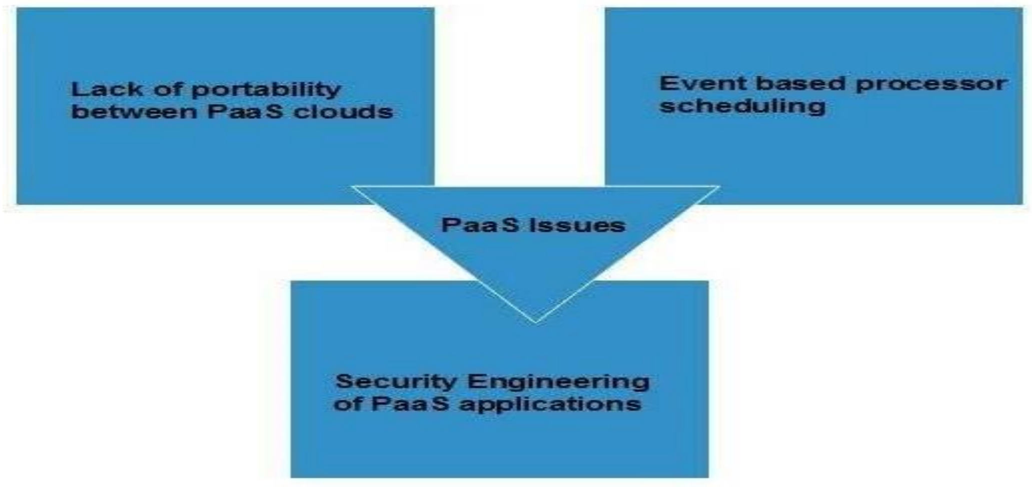
It is very easy to scale the resources up or down automatically, based on their demand.

4. More current system software

It is the responsibility of the cloud provider to maintain software versions and patch installations.

Issues:

Like **SaaS, PaaS** also places significant burdens on customer's browsers to maintain reliable and secure connections to the provider’s systems. Therefore, PaaS shares many of the issues of SaaS. However, there are some specific issues associated with PaaS as shown in the following diagram:


Lack of portability between PaaS clouds
Event based processor scheduling
PaaS Issues
Security Engineering of PaaS applications

1. Lack of portability between PaaS clouds

Although standard languages are used, yet the implementations of platform services may vary. For example, file, queue, or hash table interfaces of one platform may differ from another, making it difficult to transfer the workloads from one platform to another.

2. Event based processor scheduling

The PaaS applications are event-oriented which poses resource constraints on applications, i.e., they have to answer a request in a given interval of time.

3. Security engineering of PaaS applications

Since PaaS applications are dependent on network, they must explicitly use cryptography and manage security exposures.

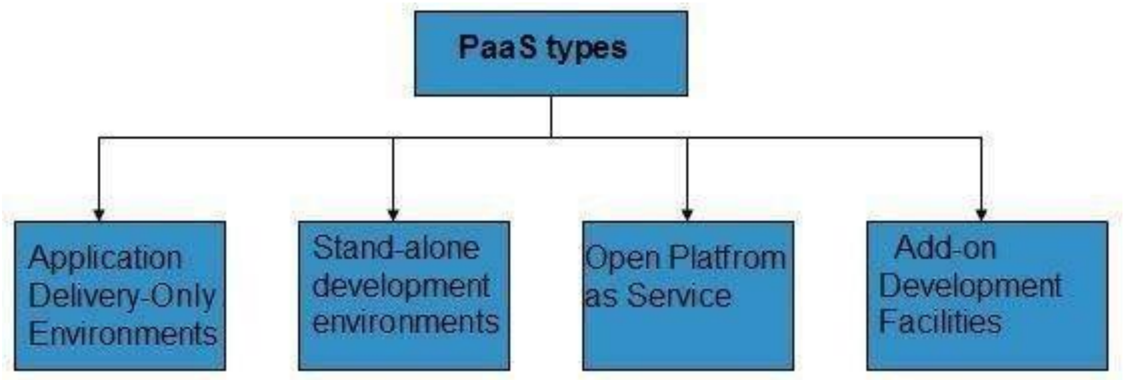
Characteristics

Here are the characteristics of PaaS service model:

- PaaS offers browser based development environment. It allows the developer to create database and edit the application code either via Application Programming Interface or point-and-click tools.
- PaaS provides built-in security, scalability, and web service interfaces.
- PaaS provides built-in tools for defining workflow, approval processes, and business rules.
- It is easy to integrate PaaS with other applications on the same platform.
- PaaS also provides web services interfaces that allow us to connect the applications outside the platform.

PaaS Types

Based on the functions, PaaS can be classified into four types as shown in the following diagram:


PaaS types
Application Delivery-Only Environments
Stand-alone development environments
Open Platfrom as Service
Add-on Development Facilities

1. Stand-alone development environments

The stand-alone PaaS works as an independent entity for a specific function. It does not include licensing or technical dependencies on specific SaaS applications.

2. Application delivery-only environments

The application delivery PaaS includes on-demand scaling and application security.

3. Open platform as a service

Open PaaS offers an open source software that helps a PaaS provider to run applications.

4. Add-on development facilities

The add-on PaaS allows to customize the existing SaaS platform.

2.3 Software-as-a-Service (SaaS):

Software-as–a-Service (SaaS) model allows to provide software application as a service to the end users. It refers to a software that is deployed on a host service and is accessible via Internet. There are several SaaS applications listed below:

- Billing and invoicing system
- Customer Relationship Management (CRM) applications
- Help desk applications
- Human Resource (HR) solutions

Some of the SaaS applications are not customizable such as Microsoft Office Suite. But SaaS provides us Application Programming Interface (API), which allows the developer to develop a customized application.

Characteristics

Here are the characteristics of SaaS service model:

- SaaS makes the software available over the Internet.
- The software applications are maintained by the vendor.
- The license to the software may be subscription based or usage based. And it is billed on recurring basis.
- SaaS applications are cost-effective since they do not require any maintenance at end user side.
- They are available on demand.
- They can be scaled up or down on demand.
- They are automatically upgraded and updated.
- SaaS offers shared data model. Therefore, multiple users can share single instance of infrastructure. It is not required to hard code the functionality for individual users.
- All users run the same version of the software.

Benefits

Using SaaS has proved to be beneficial in terms of scalability, efficiency and performance. Some of the benefits are listed below:

- Modest software tools
- Efficient use of software licenses
- Centralized management and data
- Platform responsibilities managed by provider
- Multitenant solutions

1. **Modest software tools**

The SaaS application deployment requires a little or no client side software installation, which results in the following benefits:

- No requirement for complex software packages at client side
- Little or no risk of configuration at client side
- Low distribution cost

2. **Efficient use of software licenses**

The customer can have single license for multiple computers running at different locations which reduces the licensing cost. Also, there is no requirement for license servers because the software runs in the provider's infrastructure.

3. **Centralized management and data**

The cloud provider stores data centrally. However, the cloud providers may store data in a decentralized manner for the sake of redundancy and reliability.

4. **Platform responsibilities managed by providers**

All platform responsibilities such as backups, system maintenance, security, hardware refresh, power management, etc. are performed by the cloud provider. The customer does not need to bother about them.

5. **Multitenant solutions**

Multitenant solutions allow multiple users to share single instance of different resources in virtual isolation. Customers can customize their application without affecting the core functionality.

Issues

There are several issues associated with SaaS, some of them are listed below:

- Browser based risks
- Network dependence

- Lack of portability between SaaS clouds

1. **Browser based risks**

If the customer visits malicious website and browser becomes infected, the subsequent access to SaaS application might compromise the customer's data.

To avoid such risks, the customer can use multiple browsers and dedicate a specific browser to access SaaS applications or can use virtual desktop while accessing the SaaS applications.

2. **Network dependence**

The SaaS application can be delivered only when network is continuously available. Also network should be reliable but the network reliability cannot be guaranteed either by cloud provider or by the customer.

3. **Lack of portability between SaaS clouds**

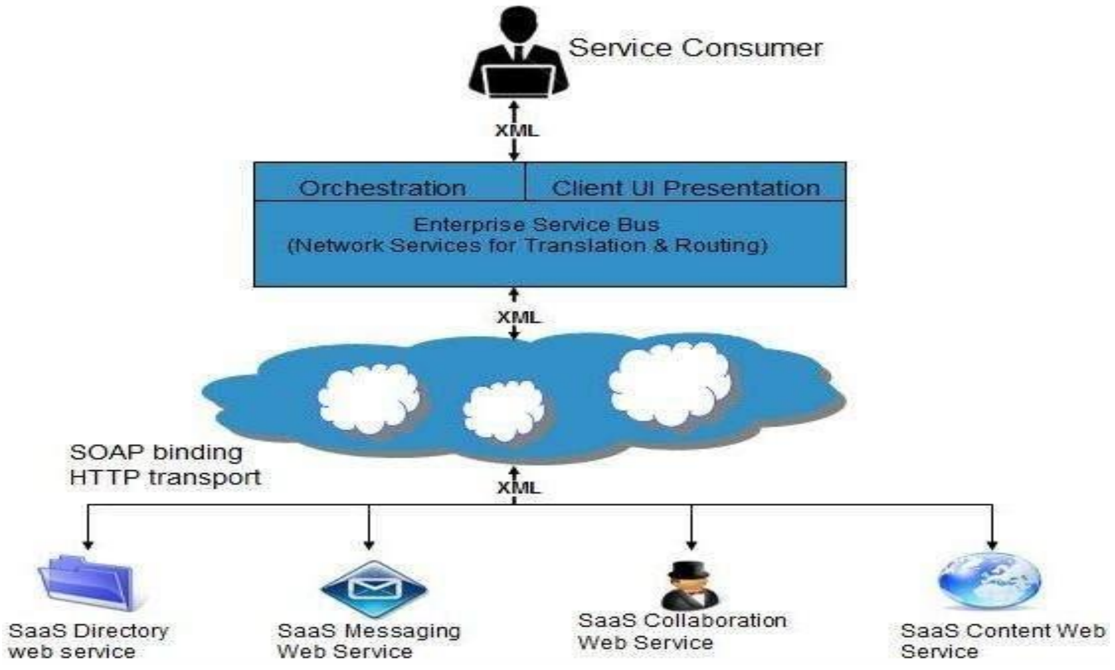
Transferring workloads from one SaaS cloud to another is not so easy because work flow, business logics, user interfaces, support scripts can be provider specific.

Open SaaS and SOA

Open SaaS uses those SaaS applications, which are developed using open source programming language. These SaaS applications can run on any open source operating system and database. Open SaaS has several benefits listed below:

- No License Required
- Low Deployment Cost
- Less Vendor Lock-in
- More portable applications
- More Robust Solution

The following diagram shows the SaaS implementation based on SOA:



SOA presents services for solution logic in an architectural model. By having these services as the foremost method of sending solutions, SOA's goal is to be more efficient, productive, and agile than other technology solutions. It provides support for realizing the advantages of computing and principles that are service-oriented. SOA implementations are made of various products, technologies, programming interfaces (for applications), and other different extensions. Application of the service-orientation principles to the software solutions will produce services. These services are the basic logic unit in SOA. Although the services have the ability to exist automatically, they're by no means isolated. There are certain standard and common features that are maintained by the services, but they have the ability to be extended and evolved independently. It's possible to combine services, enabling other services to be created. There are autonomous messages that are used for services to communicate and these messages are intelligent enough so that they can self-govern the parts of their own logic. The most important principles of SOA are service contract, autonomy, reusability, composability, loose coupling, discoverability, and statelessness.

Jericho Cloud Cube Model

Cloud computing offers a huge possibility for scalability, at almost instantaneous availability and low cost. Business managers requires IT operations to assess the risks and benefit this representation of computing model. The Jericho forum is an independent group of international information security leaders, have added their input as to how to collude securely in the clouds. The Jericho Cloud Cube Model portrays the multidimensional elements of cloud computing, that frames not only cloud use cases but also how they are set up and used.

Objectives of Jericho Forum:

The Jericho Forum's objectives associated to cloud computing are unique – "enabling secure combination in the appropriate cloud formations suited best to the business needs".

The Jericho forum:

- points out that in clouds not everything is best implemented; it may be best to conduct some business functions using a conventional non-cloud approach
- Explains the Jericho forum identified different cloud formations.
- describes benefits, key characteristics, and risks of each cloud formation
- Provide a framework for seeking in more detail the nature of different cloud formations and the issues that demands to answer to make secure places to work in and make them safe.

The Jericho Forum is actively supporting solution providers and merchants to develop the missing capabilities and services to assure customers are protected from the rough association of clouds.

Protecting our Data

First, it is necessary to categorize our data so as to know what rules must be applied to protecting it:

- Its sensitivity - must it exclusively exist at specific trust levels? If so, which?
- What supervisory /compliance restrictions apply – e.g. Must it remain within your national borders?

We only can meet this requirement if we have comprehensively adopted standards for:

- a data classification model that is easy enough for all originators of data to use – for eg the G8 Traffic Light Protocol
- an associated basis for managing trust levels
- Regulated metadata that signals to "cloud security" what security must be applied to each item of data.

With consideration of what security we must apply to our data, we're in a position to take decision:

- what data and processes to migrated to the Clouds
- At what level we want to perform in the Clouds? Cloud models isolate layers of business service from one another, for example, Infrastructure, Process, Platform, Software, and Process.
- Which Cloud composition are best suited to our needs.

Cloud Formations- the cloud cube model

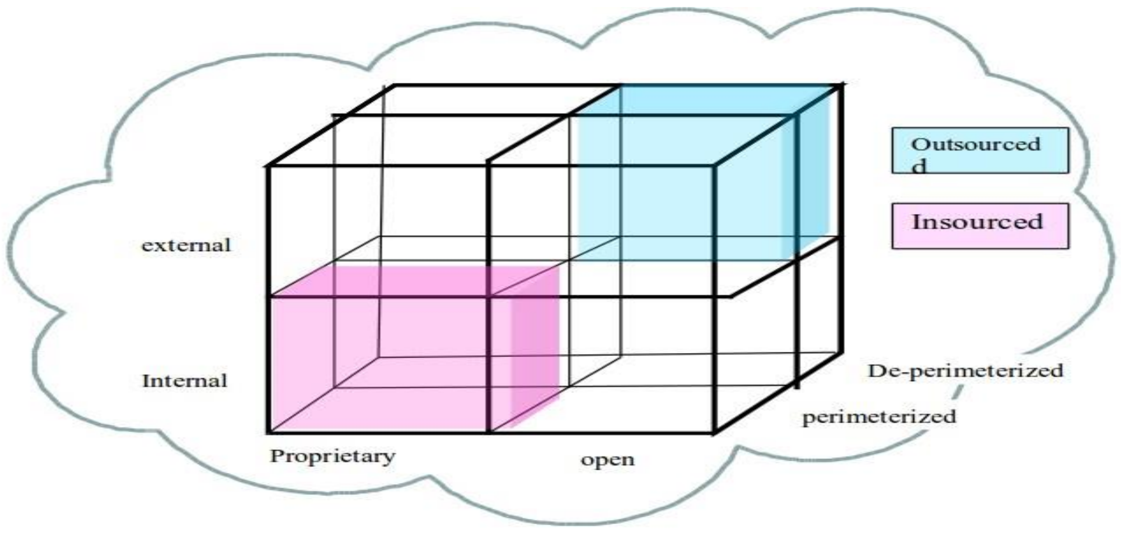


Fig: Cloud Cube Model

The Jericho forum has identified 4 gauge for judgment to differentiate cloud formations from each other and manner of their provisions. The cloud cube model summarizes these four dimensions.

Cloud Cube Model Dimensions

1. **Dimension: Internal (I) / External (E)**
2. **Dimension: Proprietary (P) / Open (O)**
3. **Dimension: Perimeterised (Per) / De-perimeterised (D-p) Architectures**
4. **Dimension: Insourced / Outsourced**

1. Dimension: Internal (I) / External (E):

This dimension defines the physical location of the data such as where does the cloud form we want to use lie, inside or outside your organization’s boundaries.

- It is Internal If it is within your own physical boundary.
- It is External If it is not within your own physical boundary.

For example, while Amazon SC33 would be external at some location “off-site, virtualized hard disks in an organization’s data center will be internal.

2. Dimension: Proprietary (P) / Open (O)

This is the dimension that represents the state of ownership of the cloud technology, interfaces, services, etc. It indicates the degree of interoperability, allowing “data/application transportability” between other cloud forms and your own systems, and the ability to pull out your data from a cloud form or to migrate it to another without force.

- Proprietary means that the organization that provides the service is keeping the means of arrangement under their ownership. As a result, when functioning in clouds that are proprietary, you may not be allowed to move to another cloud provider without specific effort or investment. Mostly the more innovative technology progress occur in the proprietary realm. As such the proprietor may choose to accomplish restrictions through patents and by keeping the involved technology a trade secret.
- Open clouds use technology that is not proprietary, meaning that there are likely to be more suppliers, and user are not as strained in being able to share your data and using the same open technology collide with selected parties. Open services tend to be those that are consumerized and widespread, and apparently most likely a published open standard, for example, email (SMTP).

3. Dimension: Perimeterised (Per) / De-perimeterised (D-p) Architectures

Architectural mindset is represented in the third dimension. De-parameterisation has always akin to the gradual failure, collapse, shrinking, and removal of the traditional IT perimeter based in silo.

- Perimeterised indicates continuing to operate within the classical IT perimeter, often indicated by "network firewalls". This approach discourages collaboration. When operating in the parameterised areas, you may simply prolong your own organization's perimeter using a VPN and operating the virtual server in your own IP domain into the external cloud computing domain, using your own directory services for access control. Then, when the computing task is finished you can pull out your perimeter back to its original classical position.
- De-perimeterised, considers that the system perimeter is architected succeeding the principles put forward in the Jericho Forum's Commandments and collusion Oriented Architectures Framework. In a de-perimeterised frame the data would be wrapped with meta-data and mechanisms that from inappropriate usage would protect the data .In a de-perimeterised environment an organization can associate securely with selected parties globally over COA capable network.

4. Dimension: Insourced / Outsourced

We define a 4th dimension having 2 states in every one of the 8 cloud forms: Per(IP,IO,EP,EO) and D-p(IP,IO,EP,EO), that answers to the question "Who do you want running our Clouds?":

- Outsourced: In outsourced dimension, the service is provided by a 3rd party
- Insourced: In insourced dimension the service is provided by your own staff under your control

These 2 states express who is managing the delivery of the cloud service(s) that you are using. This is generally a policy issue (i.e. not a technical or architectural decision but a business decision) which must be represented in a contract with the cloud service provider.

Service Oriented Architectures (SOA):

A cloud has some key characteristics: elasticity, self-service provisioning, standards based interfaces, and pay as you go. This type of functionality has to be engineered into the software. To accomplish this type of engineering requires that the foundation for the cloud be well designed and well architected. What about cloud architecture makes this approach possible? The fact is that the services and structure behind the cloud should be based on a modular architectural approach. A modular, component-based architecture enables flexibility and reuse. A Service Oriented Architecture (SOA) is what lies beneath this flexibility.

SOA is much more than a technological approach and methodology for creating IT systems. It's also a business approach and methodology. Companies have used the principles of SOA to deepen the understanding between the business and IT and to help business adapt to change.

One of the key benefits of a service oriented approach is that software is designed to reflect best practices and business processes instead of making the business operate according to the rigid structure of a technical environment. A service-oriented architecture is essentially a collection of services. A service is, in essence, a function that is well defined, self-contained, and does not depend on the context or state of other services. Services most often reflect logical business activities. Some means of connecting services to each other is needed, so services communicate with each other, have an interface, and are message-oriented. The communication between services may involve simple data passing or may require two or more services coordinating an activity. The services generally communicate using standard protocols, which allows for broad interoperability. SOA encompasses legacy systems and processes, so the effectiveness of existing investments is preserved. New services can be added or created without affecting existingservices. Service-oriented architectures are not new. The first service-oriented architectures are usually considered to be the Distributed Component Object Model (DCOM) or Object Request Brokers (ORBs), which were based on the Common Object Requesting Broker Architecture (CORBA) specification. The introduction of SOA provides a platform for technology and businessunits to meet business requirements of the modern enterprise. With SOA, your organization can use existing application systems to a greater extent and may respond faster to change requests. These benefits are attributed to several critical elements of SOA:

1. Free-standing, independent components
2. Combined by loose coupling
3. Message (XML)-based instead of API-based
4. Physical location, etc., not important

Combining Cloud and SOA:

Cloud services benefit the business by taking the best practices and business process focus of SOA to the next level. These benefits apply to both cloud service providers and cloud service users. Cloud service providers need to architect solutions by using a service-oriented approach to deliver services with the expected levels of elasticity and scalability. Companies that architect and govern business processes with reusable service-oriented components can more easily identify which components can be successfully moved to public and private clouds. A service oriented architecture (SOA) is a software architecture for building business applications that implement business processes or services through a set of loosely coupled, black-box components orchestrated to deliver a well defined level of service. This approach lets companies leverage existing assets and create new business services that are consistent, controlled, more easily changed, and more easily managed. SOA is a business approach to designing efficient IT systems that support reuse and give the businesses the flexibility to react quickly to opportunities and threats.

Characterizing SOA :

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

- SOA is black-box component architecture. The black box lets you reuse existing businessapplications; it simply adds a fairly simple adapter to them. You don't need to know everydetail of what's inside each component; SOA hides the complexity whenever possible.
- SOA components are loosely coupled. Software components are loosely coupled if they'redesigned 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.

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. Simultaneously, 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 businessprocesses.