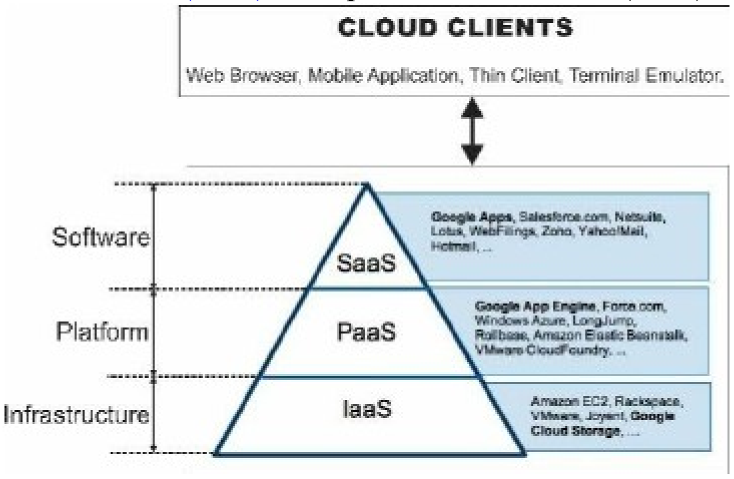
Components of Cloud Computing

Cloud Computing

A **method for delivering information technology (IT) services** in which **resources are retrieved from the Internet through web-based tools and applications**, as opposed to a direct connection to a server Rather than keeping files on a proprietary hard drive or local storage device, cloud-based storage makes it possible to save them to a remote database. As long as an electronic device has access to the web, it has access to the data and the software programs to run it.

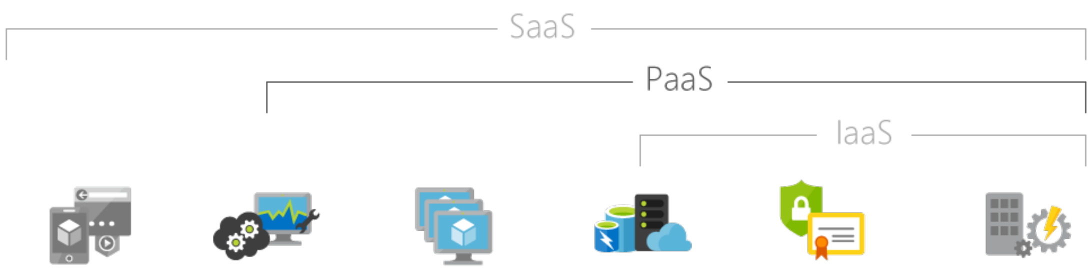
Cloud computing is not a single piece of technology, like a microchip or a cell phone. Rather, it's a system, primarily comprised of three services: infrastructure as a service (IaaS), software as a service (SaaS)+ and platform as a service (PaaS).


CLOUD CLIENTS
Web Browser, Mobile Application, Thin Client, Terminal Emulator.
Software
Platform
Infrastructure
SaaS
PaaS
IaaS
Google Apps, Salesforce.com, Netsuite, Lotus, WebFilings, Zoho, Yahoo!Mail, Hotmail, ...
Google App Engine, Force.com, Windows Azure, LongJump, Rollbase, Amazon Elastic Beanstalk, VMware CloudFoundry, ...
Amazon EC2, Rackspace, VMware, Joyent, Google Cloud Storage, ...

Software as a Service (SaaS): SaaS involves the licensure of a software application to customers. Licenses are typically provided through a pay-as-you-go model or on-demand.

Infrastructure as a Service (IaaS): Infrastructure as a service involves a method for delivering everything from operating systems to servers and storage through IP-based connectivity as part of an on-demand service. Clients can avoid the need to purchase software or servers, and instead procure these resources in an outsourced, on-demand service.

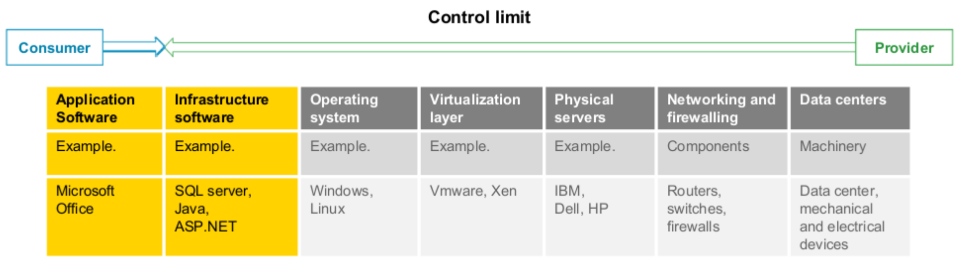
Platform as a Service (PaaS): Of the three layers of cloud-based computing, PaaS is considered the most complex. PaaS shares some similarities with SaaS, the primary difference being that instead of delivering software online, it is actually a platform for creating software that is delivered via the internet.



What is SaaS?

Software as a service is a **software licensing and delivery model** in which software is licensed on a subscription basis and is centrally hosted. Users can access it with the help of web browsers.

SaaS normally refers to **a subscription based model** where the software is **hosted in the cloud** and **accessed via the internet**.
Example: Office 365, Google Apps, Salesforce, Citrix GoToMeeting, Cisco WebEx and Netflix.


Control limit
Consumer
Provider

Application Software	Infrastructure software	Operating system	Virtualization layer	Physical servers	Networking and firewalling	Data centers
Example.	Example.	Example.	Example.	Example.	Components	Machinery
Microsoft Office	SQL server, Java, ASP.NET	Windows, Linux	Vmware, Xen	IBM, Dell, HP	Routers, switches, firewalls	Data center, mechanical and electrical devices

It is associated with the **application service providers (ASPs)** which provides **“shrink-wrap” applications** to business users **over the Internet**.

SaaS applications are **single-instance**, **multi-tenant** architecture that provide a feature-rich experience competitive with on-premise applications. Aggregator bundles SaaS offerings from different vendors and offers them as part of a unified application platform.

Multi-tenant
A single instance of software runs on a server and serves multiple tenants. A tenant is a group of users who share a common access with specific privileges to the software instance.

The **SaaS provider hosts** the application and data centrally—deploying patches . They upgrade to the application transparently, delivering access to end users over the Internet. Many **vendors provide API's that developers** use for creating composite applications. It contains various security mechanisms for Data security during transmission and storage.

SAAS Architecture:

With this model, a single version of the application, with a single configuration is used for all customers. The application is installed on multiple machines to support scalability (called horizontal scaling). In some cases, a second version of the application is set up to offer a select group of customers with access to pre-release versions of the applications for testing purposes. In this traditional model, each version of the application is based on a unique code.

Traditional software vs. Software as a Service

Traditional software	**Software as a Service**
● Users purchase the software upfront as a package and then install it onto their computer ● Licences may be limited to the number of users and/or devices where the software can be deployed	● Users subscribe to the software, usually on a monthly basis, rather than purchase it, meaning no upfront costs ● Users can usually end their subscription when they no longer require it ● Applications are updated and used online with files saved in the cloud rather than on individual computers

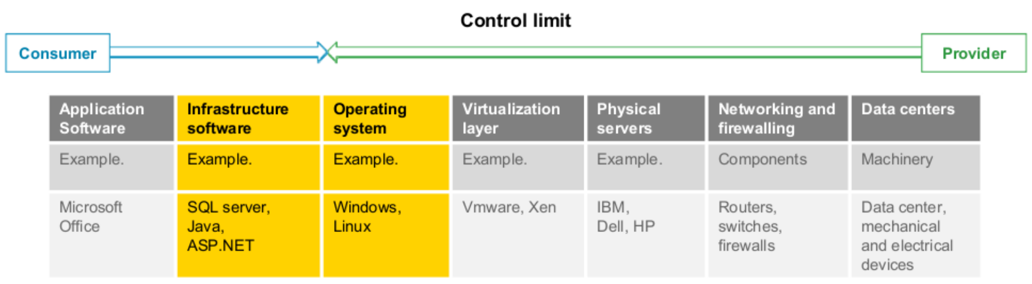
Benefits of SaaS

No hardware costs
Processing power is supplied by the cloud provider
No initial setup costs
Applications are ready to use once the user subscribes
Pay for what you use

Particularly beneficial when something is only required for a short period
Usage is scalable
Additional storage or services can be accessed on demand without needing to install new software and hardware
Updates are automatic
Updates are often free of change and deployed automatically by the software provider
Cross device compatibility
Applications can be accessed via any internet enabled device, such as desktops, smart phones and tablets
Accessible from any location
Users aren't restricted to one location and can access applications from any internet enabled device
Applications can be customised and white labelled
Applications can often be altered to suit the needs and branding of particular customers

PaaS

PaaS, or platform-as-a-service, refers to a **fully-maintained and managed environment that is hidden under a layer of abstraction** you should not even care of. The **cloud vendor takes care of maintaining the servers needed for the operations for you**, and you get **high-level databases for storing your data, services for user authentication, endpoints for client side applications etc**. This approach is much easier and faster to get up and running with, and typically satisfies most of the basic applications. You should take into consideration though, that for more complex architectures it might not be enough.


Control limit
Consumer
Provider

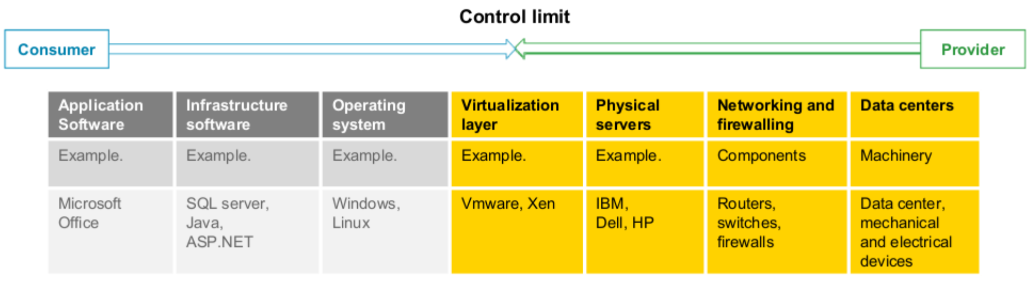
Application Software	Infrastructure software	Operating system	Virtualization layer	Physical servers	Networking and firewalling	Data centers
Example.	Example.	Example.	Example.	Example.	Components	Machinery
Microsoft Office	SQL server, Java, ASP.NET	Windows, Linux	Vmware, Xen	IBM, Dell, HP	Routers, switches, firewalls	Data center, mechanical and electrical devices

Examples: AWS Elastic Beanstalk, Windows Azure, Heroku,

Force.com, Google App Engine, Apache Stratos.

IaaS

IaaS, or infrastructure-as-a-service, refers to a **low-level solution**, like **providing a Linux Ubuntu server with nothing installed on it**. This kind of solutions is **suitable for more advanced developers who have experience with designing, configuring and securing servers infrastructure in all aspects**. IaaS services provide you with flexibility and scalability down the road, and this will most likely be the way to go when designing application for scale. This approach requires, as already mentioned before, at least one developer in your startup who has this skill-set, otherwise, your product will turn into a big mess sooner than later.



Application Software	Infrastructure software	Operating system	Virtualization layer	Physical servers	Networking and firewalling	Data centers
Example.	Example.	Example.	Example.	Example.	Components	Machinery
Microsoft Office	SQL server, Java, ASP.NET	Windows, Linux	Vmware, Xen	IBM, Dell, HP	Routers, switches, firewalls	Data center, mechanical and electrical devices

Examples: AWS Elastic Beanstalk, Windows Azure, Heroku,

Force.com, Google App Engine, Apache Stratos.