CHAPTER 1

Cloud Computing Fundamentals
William Voorsluys, James Broberg, and Rajkumar Buyya, Borko Furht

Prepared by: Dr. Faramarz Safi
Islamic Azad University, Najafabad Branch, Esfahan, Iran.
Introduction

Cloud computing can be defined as a new style of computing in which dynamically scalable and often virtualized resources are provided as a services over the Internet. Cloud computing has become a significant technology trend, and many experts expect that cloud computing will reshape information technology (IT) processes and the IT marketplace. With the cloud computing technology, users use a variety of devices, including PCs, laptops, smartphones, and PDAs to access programs, storage, and application-development platforms over the Internet, via services offered by cloud computing providers. Advantages of the cloud computing technology include cost savings, high availability, and easy scalability.
Introduction

Figure 1.1, adapted from Voas and Zhang (2009), shows six phases of computing paradigms, from dummy terminals/mainframes, to PCs, networking computing, to grid and cloud computing.

In phase 1, many users shared powerful mainframes using dummy terminals.

In phase 2, stand-alone PCs became powerful enough to meet the majority of users’ needs.

In phase 3, PCs, laptops, and servers were connected together through local networks to share resources and increase performance.

In phase 4, local networks were connected to other local networks forming a global network such as the Internet to utilize remote applications and resources.

In phase 5, grid computing provided shared computing power and storage through a distributed computing system.

In phase 6, cloud computing further provides shared resources on the Internet in a scalable and simple way.

Comparing these six computing paradigms, it looks like that cloud computing is a return to the original mainframe computing paradigm. However, these two paradigms have several important differences. Mainframe computing offers finite computing power, while cloud computing provides almost infinite power and capacity.

In addition, in mainframe computing, dummy terminals acted as user interface devices, while in cloud computing powerful PCs can provide local computing power and caching support.

Fig. 1.1 Six computing paradigms – from mainframe computing to Internet computing, to grid computing and cloud computing (adapted from Voas and Zhang (2009))
Enabling Technologies
Convergence of Various Advances Leading to the Advent of Cloud Computing
Enabling Technologies: virtualization

• The advantage of cloud computing is the ability to virtualize and share resources among different applications with the objective for better server utilization.

• In non-cloud computing three independent platforms exist for three different applications running on its own server.

• In the cloud, servers can be shared, or virtualized, for operating systems and applications resulting in fewer servers (in specific example two servers).

Fig. 1.6 An example of virtualization: in non-cloud computing there is a need for three servers; in the cloud computing, two servers are used (adapted from Jones)
Enabling Technologies

Virtualization

- The idea of virtualizing a computer system’s resources, including processors, memory, and I/O devices has been well established for decades, aiming at improving, sharing and utilization of computer systems.
- Hardware virtualization allows running multiple operating systems and software stacks on a single physical platform. A software layer, the virtual machine monitor (VMM), also called a hypervisor, mediates access to the physical hardware presenting to each guest operating system a virtual machine (VM), which is a set of several innovative technologies multi-core chips, Para-virtualization, hardware-assisted virtualization, and live migration of VMs has contributed to an increasing adoption of virtualization on server systems.
- Researches and practitioners have been emphasizing three basic capabilities regarding management of workload in a virtualized system, namely isolation, consolidation, and migration.
- Workload isolation is achieved since all program instructions are fully confined inside a VM, which leads to improvements in security.
- Better reliability is also achieved because software failures inside one VM do not affect others.
- Moreover, better performance control is attained since execution of one VM should not affect the performance of another VM.
Enabling Technologies
Virtualization

• The consolidation of several individual and heterogeneous workloads onto a single physical platform leads to better system utilization.

• This practice is also employed for overcoming potential software and hardware incompatibilities in case of upgrades, given that it is possible to run legacy and new operation systems concurrently.

• Workload migration, also referred to as application mobility, targets at facilitating hardware maintenance, load balancing, and disaster recovery. It is done by encapsulating a guest OS state within a VM and allowing it to be suspended, fully serialized, migrated to a different platform, and resumed immediately or preserved to be restored at a later date [22].

• A VM’s state includes a full disk or partition image, configuration files, and an image of its RAM.
Enabling Technologies

virtualization

A number of VM platforms exist that are the basis of many utility or cloud computing environments. The most notable ones, VMWare, Xen, and KVM, are outlined in the following sections.

• VMWare ESXi is a pioneer in the virtualization market. Its ecosystem of tools ranges from server and desktop virtualization to high-level management tools. ESXi is a VMM from VMWare. It provides advanced virtualization techniques of processor, memory, and I/O. Especially, through memory ballooning and page sharing, it can overcommit memory, thus increasing the density of VMs inside a single physical server.

• Xen. The 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, on which the guest operating system, by means of a specialized kernel, can interact with the hypervisor, thus significantly improving performance. In addition to an open-source distribution, Xen currently forms the base of commercial hypervisors of a number of vendors, most notably Citrix Xen Server and Oracle VM.

• KVM. The kernel-based virtual machine (KVM) is a Linux virtualization subsystem. It has been part of the mainline Linux kernel since version 2.6.20, thus being natively supported by several distributions. In addition, activities such as memory management and scheduling are carried out by existing kernel features, thus making 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, it supports several versions of Windows, Linux, and UNIX.
Enabling Technologies
Web service and service-oriented architecture, service flows and workflows, and Web 2.0 and mash-up.

**Web Service and Service Oriented Architecture:**

- **Web services (WS) open standards** has significantly contributed to advances in the domain of software integration. Web services can 1) glue together applications running on different messaging product platforms, 2) enabling information from one application to be made available to others, and 3) enabling internal applications to be made available over the Internet.

- A rich WS software stack has been specified and standardized, resulting in a multitude of technologies to describe, compose, and orchestrate services, package and transport messages between services, publish and discover services, represent quality of service (QoS) parameters, and ensure security in service access.

- WS standards have been created on top of existing ubiquitous technologies such as HTTP and XML, thus providing a common mechanism for delivering services. The purpose of a SOA is to address requirements of **loosely coupled, standards-based, and protocol-independent distributed computing**.

- In a SOA, software resources are packaged as “services,” that provide standard business functionality and are independent of the state or context of other services. Services are described in a standard definition language (**WSDL**) and have a published interface (**UDDI**).

- The advent of Web 2.0. information and services may be programmatically aggregated, acting as building blocks of complex compositions, called **service mashups (Web Service Composition)**. i.e. an enterprise application that follows the SOA paradigm is a collection of services that together perform complex business logic.
Enabling Technologies
Web service and service-oriented architecture, service flows and workflows, and Web 2.0 and mash-up.

• Many service providers, such as Amazon, del.icio.us, Facebook, and Google, make their service APIs publicly accessible using standard protocols such as SOAP and REST [14]. Consequently, one can put an idea of a fully functional Web application into practice just by gluing pieces with few lines of code.

• For example, programmable Web-2 is a public repository of service APIs and mashups currently listing thousands of APIs and mashups. Popular APIs such as Google Maps, Flickr, YouTube, Amazon e-Commerce, and Twitter, when combined, produce a variety of interesting solutions, from finding video game retailers to weather maps. Similarly, Salesforce.com’s offers AppExchange, which enables the sharing of solutions developed by third-party developers on top of Salesforce.com components.

• In the **Software as a Service (SaaS)** domain, cloud applications can be built as compositions of other services from the same or different providers. Services such as user authentication, e-mail, payroll management, and calendars are examples of building blocks that can be reused and combined in a business solution.
Enabling Technologies

Grid Computing

• Grid computing enables aggregation of distributed resources and transparently access to them. Most production grids such as TeraGrid and EGEE seek to share compute and storage resources distributed across different administrative domains, with their main focus being speeding up a broad range of scientific applications, such as climate modeling, drug design, and protein analysis.

• A key aspect of the grid vision realization has been building standard Web services-based protocols that allow distributed resources to be “discovered, accessed, allocated, monitored, accounted for, and billed for, etc., and in general managed as a single virtual system.”

• The Open Grid Services Architecture (OGSA) addresses this need for standardization by defining a set of core capabilities and behaviors that address key concerns in grid systems.

• Globus Toolkit is a middleware that implements several standard Grid services and over the years has aided the deployment of several service-oriented Grid infrastructures and applications. An ecosystem of tools is available to interact with service grids, including grid brokers, which facilitate user interaction with multiple middleware and implement policies to meet QoS needs. The development of standardized protocols for several grid computing activities has contributed—theoretically—to allow delivery of on-demand computing services over the Internet.
Enabling Technologies
Grid Computing

However, ensuring QoS in grids has been perceived as a difficult endeavor.

- **Lack of performance isolation** has prevented grids adoption in a variety of scenarios, especially on environments where resources are oversubscribed or users are uncooperative. Activities associated with one user or virtual organization (VO) can influence, in an uncontrollable way, the performance perceived by other users using the same platform. Therefore, the impossibility of enforcing QoS and guaranteeing execution time became a problem, especially for time-critical applications.

- **Another issue** that has lead to frustration when using grids is the availability of resources with diverse software configurations, including disparate operating systems, libraries, compilers, runtime environments, and so forth. At the same time, user applications would often run only on specially customized environments.

- Consequently, a portability barrier has often been present on most grid infrastructures, inhibiting users of adopting grids as utility computing environments.

- **Virtualization technology is the perfect fit to issues** that have caused frustration when using grids, such as hosting many dissimilar software applications on a single physical platform. In this direction, some research projects (e.g., Globus Virtual Workspaces) aimed at evolving grids to support an additional layer to virtualize computation, storage, and network resources.
Enabling Technologies
Utility Computing

• With increasing popularity and usage, large grid installations have faced new problems, such as excessive demands for resources coupled with strategic and adversarial behavior by users.

• Initially, grid resource management techniques did not ensure fair and equitable access to resources in many systems. Traditional metrics (throughput, waiting time, and slowdown) failed to capture the more subtle requirements of users. There were no real incentives for users to be flexible about resource requirements or job deadlines, nor provisions to accommodate users with urgent work.

• In utility computing environments, users assign a “utility” value to their jobs, where utility is a fixed or time-varying valuation that captures various QoS constraints (deadline, importance, satisfaction). The valuation is the amount they are willing to pay a service provider to satisfy their demands. The service providers then attempt to maximize their own utility, it means utility may directly correlate with their profit.
Enabling Technologies
Autonomic Computing

- The increasing complexity of computing systems has motivated research on autonomic computing, which seeks to improve systems by decreasing human involvement in their operation. In other words, systems should manage themselves, with high-level guidance from humans.

- Autonomic, or self-managing, systems rely on monitoring probes and gauges (sensors), on an adaptation engine (autonomic manager) for computing optimizations based on monitoring data, and on effectors to carry out changes on the system. IBM’s Autonomic Computing Initiative has contributed to define the four properties of autonomic systems: self-configuration, self-optimization, self-healing, and self-protection. IBM has also suggested a reference model for autonomic control loops of autonomic managers, called MAPE-K (Monitor Analyze Plan Execute-Knowledge).

- The large data centers of cloud computing providers must be managed in an efficient way. In this sense, the concepts of autonomic computing inspire software technologies for data center automation, which may perform tasks such as: management of service levels of running applications; management of data center capacity; proactive disaster recovery; and automation of VM provisioning.
Cloud Computing Features

Cloud computing brings a number of new features compared to other computing paradigms (Wang et al., 2008; Grossman, 2009). There are briefly described in this section:

- **Scalability and on-demand services:**
  Cloud computing provides resources and services for users on demand. The resources are scalable over several data centers. To support this expectation, clouds must allow self-service access so that customers can request, customize, pay, and use services without intervention of human operators.

- **User-centric interface:**
  Cloud interfaces are location independent and can be accesses by well established interfaces such as Web services and Internet browsers.

- **Guaranteed Quality of Service (QoS) or Service Level Agreement (SLA):**
  Cloud computed can guarantee QoS for users in terms of hardware/CPU performance, bandwidth, and memory capacity.

- **Autonomous system:**
  The cloud computing systems are autonomous systems managed transparently to users. However, software and data inside clouds can be automatically reconfigured and consolidated to a simple platform depending on user’s needs.
Cloud Computing Features

- **Per-usage Metering and Billing**
  Cloud computing eliminates up-front commitment by users, allowing them to request and use only the necessary amount. Services must be priced on a short term basis (e.g., by the hour), allowing users to release (and not pay for) resources as soon as they are not needed. For these reasons, clouds must implement features to allow efficient trading of service such as pricing, accounting, and billing. Metering should be done accordingly for different types of service (e.g., storage, processing, and bandwidth) and usage promptly reported, thus providing greater transparency.

- **Elasticity**
  Cloud computing gives the illusion of infinite computing resources available on demand. Therefore users expect clouds to rapidly provide resources in any quantity at any time. In particular, it is expected that the additional resources can be (a) provisioned, possibly automatically, when an application load increases and (b) released when load decreases (scale up and down).

- **Customization**
  In a multi-tenant cloud a great disparity between user needs is often the case. Thus, resources rented from the cloud must be highly customizable. In the case of infrastructure services, customization means allowing users to deploy specialized virtual appliances and to be given privileged (root) access to the virtual servers. Other service classes (PaaS and SaaS) offer less flexibility and are not suitable for general-purpose computing, but still are expected to provide a certain level of customization.
## Layers and Types of Cloud Computing

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

**FIGURE 1.3.** The cloud computing stack.
Layers and Types of Cloud Computing

- 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 (IaaS),
  2. Platform as a Service (PaaS),
  3. Software as a Service (SaaS).

- The reference model of Buyya et al. explains the role of each layer in an integrated architecture. A core middleware manages physical resources and the VMs deployed on top of them; in addition, it provides the required features (e.g., accounting and billing) to offer multi-tenant pay-as-you-go services.

- Cloud development environments are built on top of infrastructure services to offer application development and deployment capabilities. In this level, various programming models, libraries, APIs, and mashup editors enable the creation of a range of business, Web, and scientific applications. Once deployed in the cloud, these applications can be consumed by end users.
Layers and Types of Cloud Computing
Infrastructure as a Service

• Offering virtualized resources (computation, storage, and communication) on demand is known as Infrastructure as a Service (IaaS). Infrastructure services are considered to be the bottom layer of cloud computing systems.

• Amazon web services offers IaaS, which in the case of EC2 service means offering VMs with a software stack that can be customized similar to how an ordinary physical server would be customized. Users are given privileges to perform numerous activities to the server, such as: starting and stopping it, customizing it by installing software packages, attaching virtual disks to it, and configuring access permissions and firewall rules.
Layers of Cloud Computing
Platform as a Service

- It offers a higher level of abstraction to make a cloud easily programmable, known as Platform as a Service (PaaS). It provides Integrated Development Environment (IDE) including data security, backup and recovery, application hosting, and scalable architecture.

- A Cloud platform offers an environment on which developers create and deploy applications and do not necessarily need to know how many processors or how much memory that applications will be using. In addition, multiple programming models and specialized services (e.g., data access, authentication, and payments) are offered as building blocks to new applications.

- Google AppEngine, an example of PaaS, offers a scalable environment for developing and hosting Web applications, which should be written in specific programming languages such as Python or Java, and use the services’ own proprietary structured object data store. Building blocks include an in-memory object cache (memcache), mail service, instant messaging service (XMPP), image manipulation service, and integration with Google Accounts authentication service.
Layers of Cloud Computing
Platform as a Service

Fig. 1.3 The concept of Platform-as-a-Service, Zoho Creator (adapted from “Platform as a Service,” http://www.zoho.com/creator/paas.html)
Layers of Cloud Computing
Software as a Service

Applications reside on the top of the cloud stack. Services in this layer are accessed through Web Portals. Consumers are increasingly shifting from locally installed computer systems to online software services that offer the same functionality.

Traditional desktop applications such as word processing and spreadsheet can now be accessed as a service in the web. It alleviates the burden of software maintenance for customers and simplifies development and testing providers.

Salesforce.com, which relies on SaaS model, offers business productivity applications (CRM) that reside completely on their servers, allowing customers to customize and access applications on demand.
Layers of Cloud Computing
Software as a Service - According to Chappell (2008)

- Figure 1.4a shows the cloud service SaaS, where the entire application is running in the cloud. The client contains a simple browser to access the application. A well-known example of SaaS is salesforce.com.

- Figure 1.4b illustrates another type of cloud services, where the application runs on the client; however, it accesses useful functions and services provided in the cloud. An example of this type of cloud services on the desktop is Apple’s iTunes. The desktop application plays music, while the cloud service is used to purchase a new audio and video content. An enterprise example of this cloud service is Microsoft Exchange Hosted Services. On-premises Exchange Server is using added services from the cloud including spam filtering, archiving, and other functions.

- Finally, Fig. 1.4c shows a cloud platform for creating applications, which is used by developers. The application developers create a new SaaS application using the cloud platform.
Types of Cloud Computing


- There are three types of cloud computing:
  (a) public cloud, (b) private cloud, and (c) hybrid cloud

- **public cloud (or external cloud):** computing resources are dynamically provisioned over the Internet via Web applications or Web services from an off-site third-party provider. Public clouds are run by third parties, and applications from different customers are likely to be mixed together on the cloud’s servers, storage systems, and networks.

- **Private cloud (or internal cloud):** refers to cloud computing on private networks. Private clouds are built for the exclusive use of one client, providing full control over data, security, and quality of service. Private clouds can be built and managed by a company’s own IT organization or by a cloud provider.

- **A hybrid cloud environment** combines multiple public and private cloud models. Hybrid clouds introduce the complexity of determining how to distribute applications across both a public and private cloud.
Public Cloud and Infrastructure Services

- There are many examples for vendors who publicly provide infrastructure as a service. **Amazon Elastic Compute Cloud (EC2)**[4] , but the market now bristles with lots of competition like GoGrid, Joyent Accelerator, Rackspace, AppNexus, FlexiScale, and Manjrasoft Aneka.

- Amazon Elastic Compute Cloud (EC2) services can be leveraged via Web services (SOAP or REST), a Web-based AWS (Amazon Web Service) management console, or the EC2 command line tools.

- The Amazon service provides hundreds of pre-made AMIs (Amazon Machine Images) with a variety of operating systems (i.e., Linux, Open Solaris, or Windows) and pre-loaded software.

- It provides you with complete control of your computing resources and lets you run on Amazon’s computing and infrastructure environment easily.

- It also reduces the time required for obtaining and booting a new server’s instances to minutes, thereby allowing a quick scalable capacity and resources, up and down, as the computing requirements change.

- Amazon offers different instances’ size according to (a) the resources’ needs (small, large, and extra large), (b) the high CPU’s needs it provides (medium and extra large high CPU instances), and (c) high-memory instances (extra large, double extra large, and quadruple extra large instance).
Private cloud and Infrastructure Services

A private cloud aims at providing public cloud functionality, but on private resources, while maintaining control over an organization’s data and resources to meet security and governance’s requirements in an organization.

Private cloud exhibits a highly virtualized cloud data center located inside your organization’s firewall. It may also be a private space dedicated for your company within a cloud vendor’s data center designed to handle the organization’s workloads, and in this case it is called Virtual Private Cloud (VPC). Private clouds exhibit the following characteristics:

1) Allow service provisioning and compute capability for an organization’s users in a self-service manner.
2) Automate and provide well-managed virtualized environments.
3) Optimize computing resources, and servers’ utilization.
3) Support specific workloads.

There are many examples for vendors and frameworks that provide infrastructure as a service in private setups. The best-known examples are Eucalyptus and OpenNebula (which will be covered in more detail later on).

4) It is also important to highlight a third type of cloud setup named “hybrid cloud,” in which a combination of private/internal and external cloud resources exist together by enabling outsourcing of noncritical services and functions in public cloud and keeping the critical ones internal.

Hybrid cloud’s main function is to release resources from a public cloud and to handle sudden demand usage, which is called “cloud bursting.”
Types of Cloud Computing

Deployment Models

Armbrust et al. propose definitions for public cloud as a “cloud made available in a pay-as-you-go manner to the general public” and private cloud as “internal data center of a business or other organization, not made available to the general public.”

In most cases, establishing a private cloud means restructuring an existing infrastructure by adding virtualization and cloud-like interfaces. This allows users to interact with the local data center while experiencing the same advantages of public clouds, most notably self-service interface, privileged access to virtual servers, and per-usage metering and billing.

A community cloud is “shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations).”

A hybrid cloud takes shape when a private cloud is supplemented with computing capacity from public clouds.

The approach of temporarily renting capacity to handle spikes in load is known as “cloud-bursting”.
Cloud Computing Versus Cloud Services
(Jens, 2008)

Cloud computing is the IT foundation for cloud services and it consists of technologies that enable cloud services.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Infrastructure systems</td>
<td>It includes servers, storage, and networks that can scale as per user demand.</td>
</tr>
<tr>
<td>Application software</td>
<td>It provides Web-based user interface, Web services APIs, and a rich variety of configurations.</td>
</tr>
<tr>
<td>Application development and deployment software</td>
<td>It supports the development and integration of cloud application software.</td>
</tr>
<tr>
<td>System and application management software</td>
<td>It supports rapid self-service provisioning and configuration and usage monitoring.</td>
</tr>
<tr>
<td>IP networks</td>
<td>They connect end users to the cloud and the infrastructure components.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offsite. Third-party provider</td>
<td>In the cloud execution, it is assumed that third-party provides services. There is also a possibility of in-house cloud service delivery.</td>
</tr>
<tr>
<td>Accessed via the Internet</td>
<td>Services are accessed via standard-based, universal network access. It can also include security and quality-of-service options.</td>
</tr>
<tr>
<td>Minimal or no IT skill required</td>
<td>There is a simplified specification of requirements.</td>
</tr>
<tr>
<td>Provisioning</td>
<td>It includes self-service requesting, near real-time deployment, and dynamic and fine-grained scaling.</td>
</tr>
<tr>
<td>Pricing</td>
<td>Pricing is based on usage-based capability and it is fine-grained.</td>
</tr>
<tr>
<td>User interface</td>
<td>User interface include browsers for a variety of devices and with rich capabilities.</td>
</tr>
<tr>
<td>System interface</td>
<td>System interfaces are based on Web services APIs providing a standard framework for accessing and integrating among cloud services.</td>
</tr>
<tr>
<td>Shared resources</td>
<td>Resources are shared among cloud services users; however via configuration options with the service, there is the ability to customize.</td>
</tr>
</tbody>
</table>
Cloud Computing Features

**Cloud Computing Standards**

Cloud computing standards have not been yet fully developed; however, a number of existing typically lightweight, open standards have facilitated the growth of cloud computing:


<table>
<thead>
<tr>
<th>Applications</th>
<th>Communications: HTTP, XMPP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Security: OAuth, OpenID, SSL/TLS</td>
</tr>
<tr>
<td></td>
<td>Syndication: Atom</td>
</tr>
<tr>
<td>Client</td>
<td>Browsers: AJAX</td>
</tr>
<tr>
<td>Implementations</td>
<td>Offline:’ HTML5</td>
</tr>
<tr>
<td>Platform</td>
<td>Virtualization: OVF</td>
</tr>
<tr>
<td>Service</td>
<td>Solution stacks: LAMP</td>
</tr>
<tr>
<td></td>
<td>Data: XML, JSON</td>
</tr>
<tr>
<td></td>
<td>Web services: REST</td>
</tr>
</tbody>
</table>
Cloud Management

Cloud Infrastructure Management

A key challenge IaaS providers face when:

- building a cloud infrastructure is managing physical and virtual resources, namely servers, storage, and networks, in a holistic fashion.
- The orchestration of resources must be performed in a way to rapidly and dynamically provision resources to applications.

The software toolkit responsible for this orchestration is called a **virtual infrastructure manager (VIM)**. This type of software resembles a traditional operating system, but instead of dealing with a single computer, it aggregates resources from multiple computers, presenting a uniform view to user and applications. The term “cloud operating system”, “infrastructure sharing software”, and “virtual infrastructure engine.” are used to realize this toolkit. Sotomayor et al. present two categories of tools to manage clouds:

- **The first category—cloud toolkits**—includes those that “expose a remote and secure interface for creating, controlling and monitoring virtualize resources,” but do not specialize in VI management.
- Tools in **the second category—the virtual infrastructure managers**—provide advanced features such as automatic load balancing and server consolidation, but do not expose remote cloud-like interfaces.
Cloud Management
Cloud Infrastructure Management – Features and Case studies

Features:

• Virtualization Support
• Self-Service
• On-Demand Resource Provisioning
• Multiple Backend Hypervisors
• Storage Virtualization
• Interface to Public Clouds
• Virtual Networking
• Dynamic Resource Allocation
• Virtual Clusters
• Reservation and Negotiation Mechanism
• High Availability and Data Recovery

Case Studies:

• Apache VCL
• AppLogic
• Citrix Essentials
• Enomaly ECP
• Eucalyptus
• Nimbus3
• OpenNebula
• OpenPEX
• oVirt
• Platform ISF
• VMWare vSphere and vCloud
<table>
<thead>
<tr>
<th>License</th>
<th>Installation Platform of Controller</th>
<th>Client UI, API, Language Bindings</th>
<th>Backend Hypervisor(s)</th>
<th>Storage Virtualization</th>
<th>Interface to Public Cloud</th>
<th>Virtual Networks</th>
<th>Dynamic Resource Allocation</th>
<th>Advance Reservation of Capacity</th>
<th>High Availability</th>
<th>Data Protection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache VCL</td>
<td>Multi-platform (Apache/PHP)</td>
<td>Portal, XML-RPC</td>
<td>VMware ESX, ESXi, Server</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>AppLogic</td>
<td>Proprietary</td>
<td>GUI, CLI</td>
<td>Xen</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Citrix Essentials</td>
<td>Proprietary</td>
<td>GUI, CLI, Portal, XML-RPC</td>
<td>XenServer, Hyper-V</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Enomaly ECP</td>
<td>GPL v3</td>
<td>Portal, WS</td>
<td>Xen</td>
<td>No</td>
<td>Amazon EC2</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Eucalyptus</td>
<td>BSD</td>
<td>EC2 WS, CLI</td>
<td>Xen, KVM</td>
<td>No</td>
<td>EC2</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Nimbus</td>
<td>Apache v2</td>
<td>EC2 WS, WSRF, CLI</td>
<td>Xen, KVM</td>
<td>No</td>
<td>EC2</td>
<td>Yes</td>
<td>Via integration with OpenNebula</td>
<td>Yes (via integration with OpenNebula)</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>OpenNEbula</td>
<td>Apache v2</td>
<td>XML-RPC, CLI, Java</td>
<td>Xen, KVM</td>
<td>No</td>
<td>Amazon EC2, Elastic Hosts</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes (via Haizea)</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>OpenPEX</td>
<td>GPL v2</td>
<td>Portal, WS</td>
<td>XenServer</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>oVirt</td>
<td>GPL v2</td>
<td>Fedora Linux</td>
<td>Portal</td>
<td>KVM</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Platform ISF</td>
<td>Proprietary</td>
<td>Portal</td>
<td>Hyper-V, XenServer, VMware ESX</td>
<td>No</td>
<td>EC2, IBM CoD, HP Enterprise Services</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Unclear</td>
<td>Unclear</td>
</tr>
<tr>
<td>Platform VMO</td>
<td>Proprietary</td>
<td>Portal</td>
<td>XenServer</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>VMware vSphere</td>
<td>Proprietary</td>
<td>CLI, GUI, Portal, WS</td>
<td>VMware ESX, ESXi, VMware vStorage VMFS</td>
<td>No</td>
<td>VMware vCloud partners</td>
<td>Yes</td>
<td>VMware DRM</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Infrastructure as a service Providers

Features:
• Geographic Presence
• User Interface and Access to Servers
• Advanced Reservation of Capacity
• Automatic Scaling and Load Balancing
• Service-Level Agreement
• Hypervisor and Operating System Choice

Case Studies:
• Amazon Web Services
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon EC2</td>
<td>US East, Europe</td>
<td>CLI, WS, Portal</td>
<td>SSH (Linux), Remote Desktop (Windows)</td>
<td>99.95% Hour Xen</td>
<td>Linux, Windows</td>
<td>Available with Amazon CloudWatch</td>
<td>No</td>
<td>Elastic Load Balancing</td>
<td>No</td>
<td>1–20 EC2 compute units</td>
<td>1.7–15 GB; 160–1690 GB (per EBS volume)</td>
</tr>
<tr>
<td>Flexiscale</td>
<td>UK</td>
<td>Web Console</td>
<td>SSH</td>
<td>100% Hour Xen</td>
<td>Linux, Windows</td>
<td>No</td>
<td>Zeus software loadbalancing</td>
<td>Processors, memory (requires reboot)</td>
<td>No</td>
<td>1–4 CPUs</td>
<td>0.5–16 GB; 20–270 GB</td>
</tr>
<tr>
<td>GoGrid</td>
<td></td>
<td>REST, Java, PHP, Python, Ruby</td>
<td>SSH</td>
<td>100% Hour Xen</td>
<td>Linux, Windows</td>
<td>No</td>
<td>Hardware (F5)</td>
<td>No</td>
<td>1–6 CPUs</td>
<td>0.5–8 GB; 30–480 GB</td>
<td></td>
</tr>
<tr>
<td>Joyent Cloud</td>
<td>US (Emeryville, CA; San Diego, CA; Andover, MA; Dallas, TX)</td>
<td>SSH, VirtualMin (Web-based system administration)</td>
<td>No</td>
<td>100% Month OS Level (Solaris Containers)</td>
<td>OpenSolaris No</td>
<td>Both hardware (F5 networks) and software (Zeus)</td>
<td>Automatic CPU bursting (up to 8 CPUs)</td>
<td>1/16–8 CPUs</td>
<td>0.25–32 GB; 5–100 GB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rackspace Cloud Servers</td>
<td>US (Dallas, TX)</td>
<td>Portal, REST, Python, PHP, Java, C#, .NET</td>
<td>No</td>
<td>100% Hour Xen</td>
<td>Linux</td>
<td>No</td>
<td>No</td>
<td>Memory, disk (requires reboot)</td>
<td>Automatic CPU bursting (up to 100% of available CPU power of physical host)</td>
<td>0.25–16 GB; 10–620 GB</td>
<td></td>
</tr>
</tbody>
</table>

TABLE 1.2. Feature Comparison Public Cloud Offerings (Infrastructure as a Service)
Platform as a Service Providers

Features:
- Programming Models, Languages, and Frameworks
- Persistence Options

Case Studies:
- Aneka
- App Engine
- Microsoft Azure
- Force.com
- Heroku
<table>
<thead>
<tr>
<th>Platform</th>
<th>Target Use</th>
<th>Programming Language, Frameworks</th>
<th>Developer Tools</th>
<th>Programming Models</th>
<th>Persistence Options</th>
<th>Automatic Scaling</th>
<th>Backend Infrastructure Providers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aneka</td>
<td>.Net enterprise applications, HPC</td>
<td>.NET</td>
<td>Standalone SDK</td>
<td>Threads, Task, MapReduce</td>
<td>Flat files, RDBMS, HDFS</td>
<td>No</td>
<td>Amazon EC2</td>
</tr>
<tr>
<td>AppEngine</td>
<td>Web applications</td>
<td>Python, Java</td>
<td>Eclipse-based IDE</td>
<td>Request-based Web programming</td>
<td>BigTable</td>
<td>Yes</td>
<td>Own data centers</td>
</tr>
<tr>
<td>Force.com</td>
<td>Enterprise applications (esp. CRM)</td>
<td>Apex</td>
<td>Eclipse-based IDE, Web-based wizard</td>
<td>Workflow, Excel-like formula language, Request-based web programming</td>
<td>Own object database</td>
<td>Unclear</td>
<td>Own data centers</td>
</tr>
<tr>
<td>Microsoft Windows Azure</td>
<td>Enterprise and Web applications</td>
<td>.NET</td>
<td>Azure tools for Microsoft Visual Studio</td>
<td>Unrestricted</td>
<td>Table/BLOB/queue storage, SQL services</td>
<td>Yes</td>
<td>Own data centers</td>
</tr>
<tr>
<td>Heroku</td>
<td>Web applications</td>
<td>Ruby on Rails</td>
<td>Command-line tools</td>
<td>Request-based web programming</td>
<td>PostgreSQL, Amazon RDS</td>
<td>Yes</td>
<td>Amazon EC2</td>
</tr>
<tr>
<td>Amazon Elastic MapReduce</td>
<td>Data processing</td>
<td>Hive and Pig, Cascading, Java, Ruby, Perl, Python, PHP, R, C++</td>
<td>Karmasphere Studio for Hadoop (NetBeans-based)</td>
<td>MapReduce</td>
<td>Amazon S3</td>
<td>No</td>
<td>Amazon EC2</td>
</tr>
</tbody>
</table>
Cloud Computing Challenges and Risks

- Security, Privacy, and Trust
  - Current cloud offerings are essentially public...exposing the system to more attacks
  - Security and privacy affect the entire cloud computing stack, since there is a massive use of third-party services and infrastructures that are used to host important data or to perform critical operations.
  - Legal and regulatory issues also need attention.
    - When data are moved into the cloud, providers may choose to locate them anywhere on the planet.
    - The physical location of data centers determine the set of laws that can be applied to the management of data. For instance, specific cryptography techniques cannot be used in some countries. Similarly, country laws can impose that sensitive data such as patient health records, are to be stored within national borders.

- Data Lock-in and Standardization
  - Data might be locked-in by a certain provider
  - Users may want to move data and applications out from a provider to another one.
  - Current cloud computing infrastructures and platforms do not employ standard methods of storing user data and applications; therefore, data are not portable.
  - The answer to this challenge is open standards such as Unified Cloud Interface (UCI), or Open Virtual Format (OVF).
Cloud Computing Challenges and Risks

- **Availability, Fault-Tolerance, and Disaster Recovery**
  - Users expectations must be satisfied when they move to clouds. Expectations such as availability of the service, overall performance, or what will happen if something goes wrong in the system.
  - Service Level Agreements (SLAs) include QoS requirements must be set up between users and providers to act as warranty. Penalties for violating the expectations must also be approved.

- **Resource Management**
  - Physical resources such as CPU cores, disk space, and network bandwidth must be sliced and shared among virtual machines.
  - The multi-dimensional nature of VMs complicates finding a good mapping of VMs onto available physical hosts.
  - Dimensions to be considered: number of CPUs, amount of memory, size of virtual disks, and network bandwidth.
  - Dynamic VM mapping policies are easy ways of preempting low-priority allocations in favor of high-priority ones.
  - Migration of VMs, when to initiate migration? which VM? Where? Relocating datacenter load, and there is a trade-off between negative impact of migration and stability of service.
  - How much data must be travelled through space (i.e. migration), time (i.e., check-pointing and rewinding), load balancing, backup and recovery.
  - Dynamic provisioning of new VMs and replicating existing VMs require efficient mechanism to make VM block storage devices (i.e., image files) quickly available at selected hosts.
Cloud Computing Challenges and Risks

• Energy Efficiency

Data centers consume large amounts of electricity. According to a data published by HP[4], 100 server racks can consume 1.3 MW of power and another 1.3 MW are required by the cooling system, thus costing USD 2.6 million per year. Besides the monetary cost, data centers significantly impact the environment in terms of CO₂ emissions from the cooling systems [52].

In addition to optimize application performance, dynamic resource management can also improve utilization and consequently minimize energy consumption in data centers. This can be done by judiciously consolidating workload onto smaller number of servers and turning off idle resources.
Cloud Computing Platforms Pricing

Pricing for cloud platforms and services is based on three key dimensions: (i) **storage**, (ii) **bandwidth**, and (iii) **compute**.

- **Storage** is typically measured as average daily amount of data stored in GB over a monthly period.
- **Bandwidth** is measured by calculating the total amount of data transferred in and out of platform service through transaction and batch processing. Generally, data transfer between services within the same platform is free in many platforms.
- **Compute** is measured as the time units needed to run an instance, or application, or machine to servicing requests. Table 6 compares pricing for three major cloud computing platforms.

In summary, by analyzing the cost of cloud computing, depending on the application characteristics the cost of deploying an application could vary based on the selected platform. From Table 1.6, it seems that the unit pricing for three major platforms is quite similar. Besides unit pricing, it is important to translate it into monthly application development, deployments and maintenance costs.

<table>
<thead>
<tr>
<th>Resource</th>
<th>UNIT</th>
<th>Amazon</th>
<th>Google</th>
<th>Microsoft</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stored data</td>
<td>GB per month</td>
<td>$0.10</td>
<td>$0.15</td>
<td>$0.15</td>
</tr>
<tr>
<td>Storage transaction</td>
<td>Per 10 K requests</td>
<td>$0.10</td>
<td>$0.10</td>
<td>$0.10</td>
</tr>
<tr>
<td>Outgoing bandwidth</td>
<td>GB</td>
<td>$0.10 – $0.17</td>
<td>$0.12</td>
<td>$0.15</td>
</tr>
<tr>
<td>Incoming bandwidth</td>
<td>GB</td>
<td>$0.10</td>
<td>$0.10</td>
<td>$0.10</td>
</tr>
<tr>
<td>Compute time</td>
<td>Instance Hours</td>
<td>$0.10 – $1.20</td>
<td>$0.10</td>
<td>$0.12</td>
</tr>
</tbody>
</table>
Cloud Computing Platforms

Cloud Computing Components and Their Vendors

The main elements comprising cloud computing platforms include computer hardware, storage, infrastructure, computer software, operating systems, and platform virtualization. The leading vendors providing cloud computing components are shown in Table 1.7 (“Cloud Computing,” Wikipedia, http://en.wikipedia.org/wiki/Cloud_computing).

<table>
<thead>
<tr>
<th>Cloud computing components</th>
<th>Vendors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computer hardware</td>
<td>Dell, HP, IBM, Sun</td>
</tr>
<tr>
<td>Storage</td>
<td>Sun, EMC, IBM</td>
</tr>
<tr>
<td>Infrastructure</td>
<td>Cisco, Juniper Networks, Brocade Communication</td>
</tr>
<tr>
<td>Computer software</td>
<td>3tera, Eucalyptus, G-Eclipse, Hadoop</td>
</tr>
<tr>
<td>Operating systems</td>
<td>Solaris, AIX, Linux (Red Hat, Ubuntu)</td>
</tr>
<tr>
<td>Platform virtualization</td>
<td>Citrix, VMWare, IBM, Xen, Linux KVM, Microsoft, Sun xVM</td>
</tr>
</tbody>
</table>
In summary, cloud computing is definitely a type of computing paradigm/architecture that will remain for a long time to come.

In the near future, cloud computing can emerge in various directions. One possible scenario for the future is that an enterprise may use a distributed hybrid cloud.

According to this scenario, the enterprise will use the core applications on its private cloud, while some other applications will be distributed on several public/private clouds, which are optimized for specific applications.