CHAPTER 1

INTRODUCTION TO CLOUD COMPUTING

WILLIAM VOORSLUYS, JAMES BROBERG, and RAJKUMAR BUYYA

1.1 CLOUD COMPUTING IN A NUTSHELL

When plugging an electric appliance into an outlet, we care neither how electric power is generated nor how it gets to that outlet. This is possible because electricity is virtualized; that is, it is readily available from a wall socket that hides power generation stations and a huge distribution grid. When extended to information technologies, this concept means delivering useful functions while hiding how their internals work. Computing itself, to be considered fully virtualized, must allow computers to be built from distributed components such as processing, storage, data, and software resources [1].

Technologies such as *cluster*, *grid*, and now, *cloud* computing, have all aimed at allowing access to large amounts of computing power in a fully virtualized manner, by aggregating resources and offering a single system view. In addition, an important aim of these technologies has been delivering computing as a utility. Utility computing describes a business model for on-demand delivery of computing power; consumers pay providers based on usage ("pay-as-you-go"), similar to the way in which we currently obtain services from traditional public utility services such as water, electricity, gas, and telephony.

Cloud computing has been coined as an umbrella term to describe a category of sophisticated on-demand computing services initially offered by commercial providers, such as Amazon, Google, and Microsoft. It denotes a model on which a computing infrastructure is viewed as a "cloud," from which businesses and individuals access applications from anywhere in the world on demand [2]. The main principle behind this model is offering computing, storage, and software "as a service."

Cloud Computing: Principles and Paradigms, Edited by Rajkumar Buyya, James Broberg and Andrzej Goscinski

Many practitioners in the commercial and academic spheres have attempted to define exactly what "cloud computing" is and what unique characteristics it presents. Buyya et al. [2] have defined it as follows: "Cloud is a parallel and distributed computing system consisting of a collection of inter-connected and virtualised computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements (SLA) established through negotiation between the service provider and consumers."

Vaquero et al. [3] have stated "clouds are a large pool of easily usable and accessible virtualized resources (such as hardware, development platforms and/or services). These resources can be dynamically reconfigured to adjust to a variable load (scale), allowing also for an optimum resource utilization. This pool of resources is typically exploited by a pay-per-use model in which guarantees are offered by the Infrastructure Provider by means of customized Service Level Agreements."

A recent McKinsey and Co. report [4] claims that "Clouds are hardware-based services offering compute, network, and storage capacity where: Hardware management is highly abstracted from the buyer, buyers incur infrastructure costs as variable OPEX, and infrastructure capacity is highly elastic."

A report from the University of California Berkeley [5] summarized the key characteristics of cloud computing as: "(1) the illusion of infinite computing resources; (2) the elimination of an up-front commitment by cloud users; and (3) the ability to pay for use ... as needed ..."

The National Institute of Standards and Technology (NIST) [6] characterizes cloud computing as "... a pay-per-use model for enabling available, convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage, applications, services) that can be rapidly provisioned and released with minimal management effort or service provider interaction."

In a more generic definition, Armbrust et al. [5] define cloud as the "data center hardware and software that provide services." Similarly, Sotomayor et al. [7] point out that "cloud" is more often used to refer to the IT infrastructure deployed on an Infrastructure as a Service provider data center.

While there are countless other definitions, there seems to be common characteristics between the most notable ones listed above, which a cloud should have: (i) pay-per-use (no ongoing commitment, utility prices); (ii) elastic capacity and the illusion of infinite resources; (iii) self-service interface; and (iv) resources that are abstracted or virtualised.

In addition to raw computing and storage, cloud computing providers usually offer a broad range of software services. They also include APIs and development tools that allow developers to build seamlessly scalable applications upon their services. The ultimate goal is allowing customers to run their everyday IT infrastructure "in the cloud."

A lot of hype has surrounded the cloud computing area in its infancy, often considered the most significant switch in the IT world since the advent of the

Internet [8]. In midst of such hype, a great deal of confusion arises when trying to define what cloud computing is and which computing infrastructures can be termed as "clouds."

Indeed, the long-held dream of delivering computing as a utility has been realized with the advent of cloud computing [5]. However, over the years, several technologies have matured and significantly contributed to make cloud computing viable. In this direction, this introduction tracks the roots of cloud computing by surveying the main technological advancements that significantly contributed to the advent of this emerging field. It also explains concepts and developments by categorizing and comparing the most relevant R&D efforts in cloud computing, especially public clouds, management tools, and development frameworks. The most significant practical cloud computing realizations are listed, with special focus on architectural aspects and innovative technical features.

1.2 ROOTS OF CLOUD COMPUTING

We can track the roots of clouds computing by observing the advancement of several technologies, especially in hardware (virtualization, multi-core chips), Internet technologies (Web services, service-oriented architectures, Web 2.0), distributed computing (clusters, grids), and systems management (autonomic computing, data center automation). Figure 1.1 shows the convergence of technology fields that significantly advanced and contributed to the advent of cloud computing.

Some of these technologies have been tagged as hype in their early stages of development; however, they later received significant attention from academia and were sanctioned by major industry players. Consequently, a specification and standardization process followed, leading to maturity and wide adoption. The emergence of cloud computing itself is closely linked to the maturity of such technologies. We present a closer look at the technologies that form the base of cloud computing, with the aim of providing a clearer picture of the cloud ecosystem as a whole.

1.2.1 From Mainframes to Clouds

We are currently experiencing a switch in the IT world, from in-house generated computing power into utility-supplied computing resources delivered over the Internet as Web services. This trend is similar to what occurred about a century ago when factories, which used to generate their own electric power, realized that it is was cheaper just plugging their machines into the newly formed electric power grid [8].

Computing delivered as a utility can be defined as "on demand delivery of infrastructure, applications, and business processes in a security-rich, shared, scalable, and based computer environment over the Internet for a fee" [9].

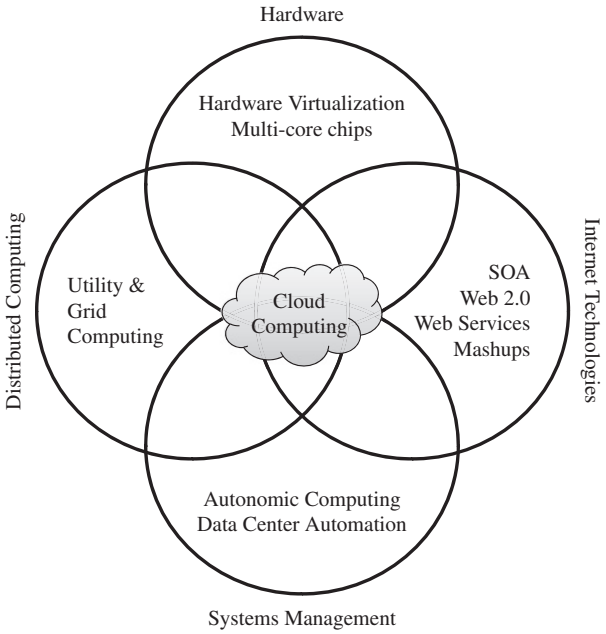

FIGURE 1.1. Convergence of various advances leading to the advent of cloud computing.

This model brings benefits to both consumers and providers of IT services. Consumers can attain reduction on IT-related costs by choosing to obtain cheaper services from external providers as opposed to heavily investing on IT infrastructure and personnel hiring. The "on-demand" component of this model allows consumers to adapt their IT usage to rapidly increasing or unpredictable computing needs.

Providers of IT services achieve better operational costs; hardware and software infrastructures are built to provide multiple solutions and serve many users, thus increasing efficiency and ultimately leading to faster return on investment (ROI) as well as lower total cost of ownership (TCO) [10].

Several technologies have in some way aimed at turning the utility computing concept into reality. In the 1970s, companies who offered common data processing tasks, such as payroll automation, operated time-shared mainframes as utilities, which could serve dozens of applications and often operated close to 100% of their capacity. In fact, mainframes had to operate at very high utilization rates simply because they were very expensive and costs should be justified by efficient usage [8].

The mainframe era collapsed with the advent of fast and inexpensive microprocessors and IT data centers moved to collections of commodity servers. Apart from its clear advantages, this new model inevitably led to isolation of workload into dedicated servers, mainly due to incompatibilities

between software stacks and operating systems [11]. In addition, the unavailability of efficient computer networks meant that IT infrastructure should be hosted in proximity to where it would be consumed. Altogether, these facts have prevented the utility computing reality of taking place on modern computer systems.

Similar to old electricity generation stations, which used to power individual factories, computing servers and desktop computers in a modern organization are often underutilized, since IT infrastructure is configured to handle theoretical demand peaks. In addition, in the early stages of electricity generation, electric current could not travel long distances without significant voltage losses. However, new paradigms emerged culminating on transmission systems able to make electricity available hundreds of kilometers far off from where it is generated. Likewise, the advent of increasingly fast fiber-optics networks has relit the fire, and new technologies for enabling sharing of computing power over great distances have appeared.

These facts reveal the potential of delivering computing services with the speed and reliability that businesses enjoy with their local machines. The benefits of economies of scale and high utilization allow providers to offer computing services for a fraction of what it costs for a typical company that generates its own computing power [8].

1.2.2 SOA, Web Services, Web 2.0, and Mashups

The emergence of Web services (WS) open standards has significantly contributed to advances in the domain of software integration [12]. Web services can glue together applications running on different messaging product platforms, enabling information from one application to be made available to others, and enabling internal applications to be made available over the Internet.

Over the years 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 [13].

WS standards have been created on top of existing ubiquitous technologies such as HTTP and XML, thus providing a common mechanism for delivering services, making them ideal for implementing a service-oriented architecture (SOA). 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," which are well-defined, self-contained modules that provide standard business functionality and are independent of the state or context of other services. Services are described in a standard definition language and have a published interface [12].

The maturity of WS has enabled the creation of powerful services that can be accessed on-demand, in a uniform way. While some WS are published with the

intent of serving end-user applications, their true power resides in its interface being accessible by other services. An enterprise application that follows the SOA paradigm is a collection of services that together perform complex business logic [12].

This concept of gluing services initially focused on the enterprise Web, but gained space in the consumer realm as well, especially with the advent of Web 2.0. In the consumer Web, information and services may be programmatically aggregated, acting as building blocks of complex compositions, called *service mashups*. 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.

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 user authentication, e-mail, payroll management, and calendars are examples of building blocks that can be reused and combined in a business solution in case a single, ready-made system does not provide all those features. Many building blocks and solutions are now available in public marketplaces. For example, Programmable Web[1] 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 eCommerce, and Twitter, when combined, produce a variety of interesting solutions, from finding video game retailers to weather maps. Similarly, Salesforce.com's offers AppExchange,[2] which enables the sharing of solutions developed by third-party developers on top of Salesforce.com components.

1.2.3 Grid Computing

Grid computing enables aggregation of distributed resources and transparently access to them. Most production grids such as TeraGrid [15] and EGEE [16] 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.

[1] http://www.programmableweb.com

[2] http://sites.force.com/appexchange

Globus Toolkit [18] 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. However, ensuring QoS in grids has been perceived as a difficult endeavor [19]. 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 [20].

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 [20].

Virtualization technology has been identified as 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 [20]) aimed at evolving grids to support an additional layer to virtualize computation, storage, and network resources.

1.2.4 Utility Computing

With increasing popularity and usage, large grid installations have faced new problems, such as excessive spikes in demand 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, where said utility may directly correlate with their profit. Providers can choose to prioritize

high yield (i.e., profit per unit of resource) user jobs, leading to a scenario where shared systems are viewed as a marketplace, where users compete for resources based on the perceived utility or value of their jobs. Further information and comparison of these utility computing environments are available in an extensive survey of these platforms [17].

1.2.5 Hardware Virtualization

Cloud computing services are usually backed by large-scale data centers composed of thousands of computers. Such data centers are built to serve many users and host many disparate applications. For this purpose, hardware virtualization can be considered as a perfect fit to overcome most operational issues of data center building and maintenance.

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 [21]. Hardware virtualization allows running multiple operating systems and software stacks on a single physical platform. As depicted in Figure 1.2, 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 virtual platform interfaces [22].

The advent of several innovative technologies—multi-core chips, paravirtualization, hardware-assisted virtualization, and live migration of VMs—has contributed to an increasing adoption of virtualization on server systems. Traditionally, perceived benefits were improvements on sharing and utilization, better manageability, and higher reliability. More recently, with the adoption of virtualization on a broad range of server and client systems, researchers and practitioners have been emphasizing three basic capabilities regarding

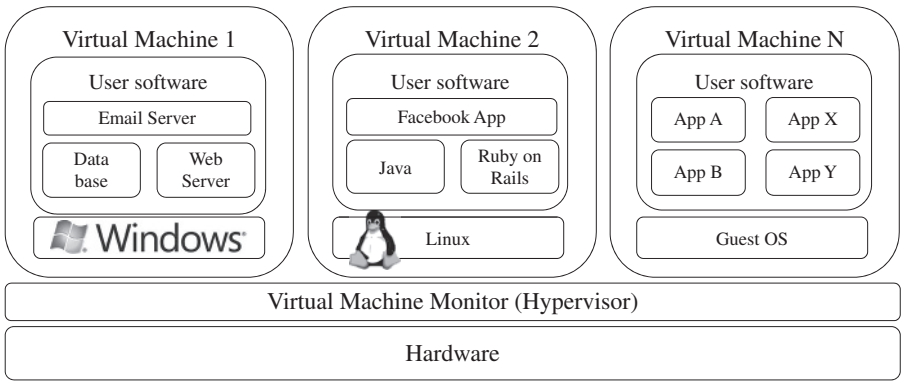

FIGURE 1.2. A hardware virtualized server hosting three virtual machines, each one running distinct operating system and user level software stack.

management of workload in a virtualized system, namely isolation, consolidation, and migration [23].

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 [22]. Moreover, better performance control is attained since execution of one VM should not affect the performance of another VM [23].

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 [22].

Workload migration, also referred to as application mobility [23], 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 [20].

A number of VMM 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. VMware is a pioneer in the virtualization market. Its ecosystem of tools ranges from server and desktop virtualization to high-level management tools [24]. ESXi is a VMM from VMWare. It is a bare-metal hypervisor, meaning that it installs directly on the physical server, whereas others may require a host operating system. 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 [25], Xen currently forms the base of commercial hypervisors of a number of vendors, most notably Citrix XenServer [26] and Oracle VM [27].

KVM. The kernel-based virtual machine (KVM) is a Linux virtualization subsystem. Is 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 [28].

KVM leverages hardware-assisted virtualization, which improves performance and allows it to support unmodified guest operating systems [29]; currently, it supports several versions of Windows, Linux, and UNIX [28].

1.2.6 Virtual Appliances and the Open Virtualization Format

An application combined with the environment needed to run it (operating system, libraries, compilers, databases, application containers, and so forth) is referred to as a "virtual appliance." Packaging application environments in the shape of virtual appliances eases software customization, configuration, and patching and improves portability. Most commonly, an appliance is shaped as a VM disk image associated with hardware requirements, and it can be readily deployed in a hypervisor.

On-line marketplaces have been set up to allow the exchange of ready-made appliances containing popular operating systems and useful software combinations, both commercial and open-source. Most notably, the VMWare virtual appliance marketplace allows users to deploy appliances on VMWare hypervisors or on partners public clouds [30], and Amazon allows developers to share specialized Amazon Machine Images (AMI) and monetize their usage on Amazon EC2 [31].

In a multitude of hypervisors, where each one supports a different VM image format and the formats are incompatible with one another, a great deal of interoperability issues arises. For instance, Amazon has its Amazon machine image (AMI) format, made popular on the Amazon EC2 public cloud. Other formats are used by Citrix XenServer, several Linux distributions that ship with KVM, Microsoft Hyper-V, and VMware ESX.

In order to facilitate packing and distribution of software to be run on VMs several vendors, including VMware, IBM, Citrix, Cisco, Microsoft, Dell, and HP, have devised the Open Virtualization Format (OVF). It aims at being "open, secure, portable, efficient and extensible" [32]. An OVF package consists of a file, or set of files, describing the VM hardware characteristics (e.g., memory, network cards, and disks), operating system details, startup, and shutdown actions, the virtual disks themselves, and other metadata containing product and licensing information. OVF also supports complex packages composed of multiple VMs (e.g., multi-tier applications) [32].

OVF's extensibility has encouraged additions relevant to management of data centers and clouds. Mathews et al. [33] have devised virtual machine contracts (VMC) as an extension to OVF. A VMC aids in communicating and managing the complex expectations that VMs have of their runtime environment and vice versa. A simple example of a VMC is when a cloud consumer wants to specify minimum and maximum amounts of a resource that a VM needs to function; similarly the cloud provider could express resource limits as a way to bound resource consumption and costs.

1.2.7 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 [34].

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) [34, 35].

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 [36].

1.3 LAYERS AND TYPES OF CLOUDS

Cloud computing services are divided into three classes, according to the abstraction level of the capability provided and the service model of providers, namely: (1) Infrastructure as a Service, (2) Platform as a Service, and (3) Software as a Service [6]. Figure 1.3 depicts the layered organization of the cloud stack from physical infrastructure to applications.

These abstraction levels can also be viewed as a layered architecture where services of a higher layer can be composed from services of the underlying layer [37]. The reference model of Buyya et al. [38] 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.

1.3.1 Infrastructure as a Service

Offering virtualized resources (computation, storage, and communication) on demand is known as Infrastructure as a Service (IaaS) [7]. A *cloud infrastructure*

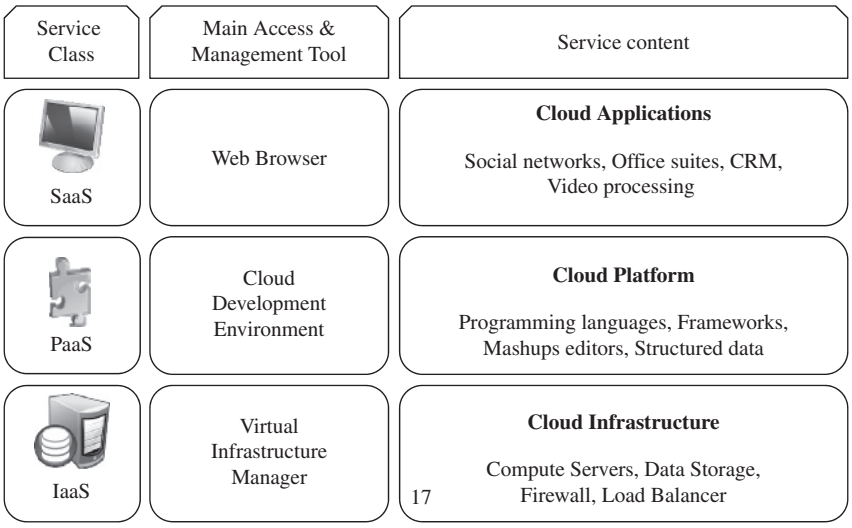

FIGURE 1.3. The cloud computing stack.

enables on-demand provisioning of servers running several choices of operating systems and a customized software stack. Infrastructure services are considered to be the bottom layer of cloud computing systems [39].

Amazon Web Services mainly offers IaaS, which in the case of its 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 firewalls rules.

1.3.2 Platform as a Service

In addition to infrastructure-oriented clouds that provide raw computing and storage services, another approach is to offer a higher level of abstraction to make a cloud easily programmable, known as Platform as a Service (PaaS). 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 [40].

Google AppEngine, an example of Platform as a Service, 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), an image manipulation service, and integration with Google Accounts authentication service.

1.3.3 Software as a Service

Applications reside on the top of the cloud stack. Services provided by this layer can be accessed by end users through Web portals. Therefore, consumers are increasingly shifting from locally installed computer programs to on-line software services that offer the same functionally. Traditional desktop applications such as word processing and spreadsheet can now be accessed as a service in the Web. This model of delivering applications, known as Software as a Service (SaaS), alleviates the burden of software maintenance for customers and simplifies development and testing for providers [37, 41].

Salesforce.com, which relies on the SaaS model, offers business productivity applications (CRM) that reside completely on their servers, allowing costumers to customize and access applications on demand.

1.3.4 Deployment Models

Although cloud computing has emerged mainly from the appearance of public computing utilities, other deployment models, with variations in physical location and distribution, have been adopted. In this sense, regardless of its service class, a cloud can be classified as public, private, community, or hybrid [6] based on model of deployment as shown in Figure 1.4.

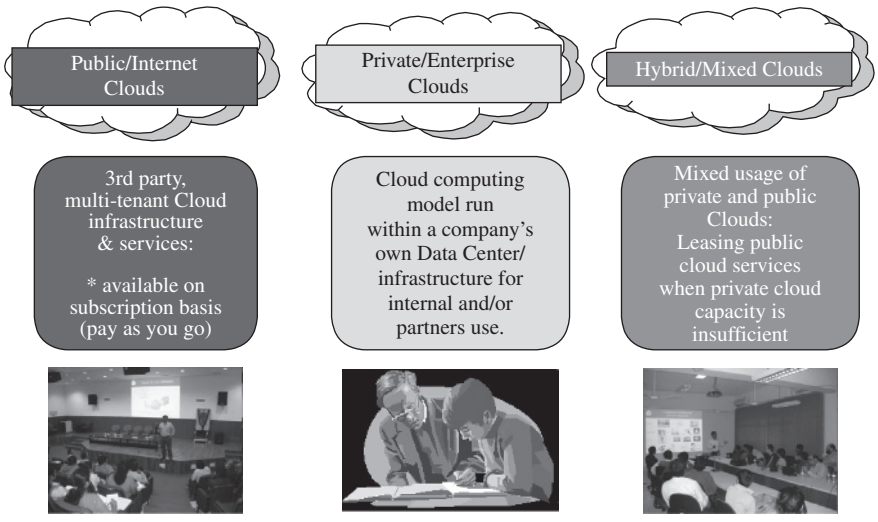

FIGURE 1.4. Types of clouds based on deployment models.

Armbrust et al. [5] 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) [6]."

A *hybrid cloud* takes shape when a private cloud is supplemented with computing capacity from public clouds [7]. The approach of temporarily renting capacity to handle spikes in load is known as "cloud-bursting" [42].

1.4 DESIRED FEATURES OF A CLOUD

Certain features of a cloud are essential to enable services that truly represent the cloud computing model and satisfy expectations of consumers, and cloud offerings must be (i) self-service, (ii) per-usage metered and billed, (iii) elastic, and (iv) customizable.

1.4.1 Self-Service

Consumers of cloud computing services expect on-demand, nearly instant access to resources. 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 [6].

1.4.2 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 [5]. For these reasons, clouds must implement features to allow efficient trading of service such as pricing, accounting, and billing [2]. Metering should be done accordingly for different types of service (e.g., storage, processing, and bandwidth) and usage promptly reported, thus providing greater transparency [6].

1.4.3 Elasticity

Cloud computing gives the illusion of infinite computing resources available on demand [5]. 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) [6].

1.4.4 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 [5], but still are expected to provide a certain level of customization.

1.5 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 [43]. The orchestration of resources must be performed in a way to rapidly and dynamically provision resources to applications [7].

The software toolkit responsible for this orchestration is called a virtual infrastructure manager (VIM) [7]. 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" is also used to refer to it [43]. Other terms include "infrastructure sharing software [44]" and "virtual infrastructure engine [45]."

Sotomayor et al. [7], in their description of the cloud ecosystem of software tools, propose a differentiation between two categories of tools used 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. However, the authors point out that there is a superposition between the categories; cloud toolkits can also manage virtual infrastructures, although they usually provide less sophisticated features than specialized VI managers do.

The availability of a remote cloud-like interface and the ability of managing many users and their permissions are the primary features that would distinguish "cloud toolkits" from "VIMs." However, in this chapter, we place both categories of tools under the same group (of the VIMs) and, when applicable, we highlight the availability of a remote interface as a feature.

Virtually all VIMs we investigated present a set of basic features related to managing the life cycle of VMs, including networking groups of VMs together and setting up virtual disks for VMs. These basic features pretty much define whether a tool can be used in practical cloud deployments or not. On the other hand, only a handful of software present advanced features (e.g., high availability) which allow them to be used in large-scale production clouds.

1.5.1 Features

We now present a list of both basic and advanced features that are usually available in VIMs.

Virtualization Support. The multi-tenancy aspect of clouds requires multiple customers with disparate requirements to be served by a single hardware infrastructure. Virtualized resources (CPUs, memory, etc.) can be sized and resized with certain flexibility. These features make hardware virtualization, the ideal technology to create a virtual infrastructure that partitions a data center among multiple tenants.

Self-Service, On-Demand Resource Provisioning. Self-service access to resources has been perceived as one the most attractive features of clouds. This feature enables users to directly obtain services from clouds, such as spawning the creation of a server and tailoring its software, configurations, and security policies, without interacting with a human system administrator. This capability "eliminates the need for more time-consuming, labor-intensive, human-driven procurement processes familiar to many in IT" [46]. Therefore, exposing a self-service interface, through which users can easily interact with the system, is a highly desirable feature of a VI manager.

Multiple Backend Hypervisors. Different virtualization models and tools offer different benefits, drawbacks, and limitations. Thus, some VI managers provide a uniform management layer regardless of the virtualization technology used. This characteristic is more visible in open-source VI managers, which usually provide pluggable drivers to interact with multiple hypervisors [7]. In this direction, the aim of libvirt [47] is to provide a uniform API that VI managers can use to manage domains (a VM or container running an instance of an operating system) in virtualized nodes using standard operations that abstract hypervisor specific calls.

Storage Virtualization. Virtualizing storage means abstracting logical storage from physical storage. By consolidating all available storage devices in a data center, it allows creating virtual disks independent from device and location. Storage devices are commonly organized in a storage area network (SAN) and attached to servers via protocols such as Fibre Channel, iSCSI, and

NFS; a storage controller provides the layer of abstraction between virtual and physical storage [48].

In the VI management sphere, storage virtualization support is often restricted to commercial products of companies such as VMWare and Citrix. Other products feature ways of pooling and managing storage devices, but administrators are still aware of each individual device.

Interface to Public Clouds. Researchers have perceived that extending the capacity of a local in-house computing infrastructure by borrowing resources from public clouds is advantageous. In this fashion, institutions can make good use of their available resources and, in case of spikes in demand, extra load can be offloaded to rented resources [45].

A VI manager can be used in a hybrid cloud setup if it offers a driver to manage the life cycle of virtualized resources obtained from external cloud providers. To the applications, the use of leased resources must ideally be transparent.

Virtual Networking. Virtual networks allow creating an isolated network on top of a physical infrastructure independently from physical topology and locations [49]. A virtual LAN (VLAN) allows isolating traffic that shares a switched network, allowing VMs to be grouped into the same broadcast domain. Additionally, a VLAN can be configured to block traffic originated from VMs from other networks. Similarly, the VPN (virtual private network) concept is used to describe a secure and private overlay network on top of a public network (most commonly the public Internet) [50].

Support for creating and configuring virtual networks to group VMs placed throughout a data center is provided by most VI managers. Additionally, VI managers that interface with public clouds often support secure VPNs connecting local and remote VMs.

Dynamic Resource Allocation. Increased awareness of energy consumption in data centers has encouraged the practice of dynamic consolidating VMs in a fewer number of servers. In cloud infrastructures, where applications have variable and dynamic needs, capacity management and demand prediction are especially complicated. This fact triggers the need for dynamic resource allocation aiming at obtaining a timely match of supply and demand [51].

Energy consumption reduction and better management of SLAs can be achieved by dynamically remapping VMs to physical machines at regular intervals. Machines that are not assigned any VM can be turned off or put on a low power state. In the same fashion, overheating can be avoided by moving load away from hotspots [52].

A number of VI managers include a dynamic resource allocation feature that continuously monitors utilization across resource pools and reallocates available resources among VMs according to application needs.

Virtual Clusters. Several VI managers can holistically manage groups of VMs. This feature is useful for provisioning computing *virtual clusters on demand*, and interconnected VMs for multi-tier Internet applications [53].

Reservation and Negotiation Mechanism. When users request computational resources to available at a specific time, requests are termed advance reservations (AR), in contrast to best-effort requests, when users request resources whenever available [54]. To support complex requests, such as AR, a VI manager must allow users to "lease" resources expressing more complex terms (e.g., the period of time of a reservation). This is especially useful in clouds on which resources are scarce; since not all requests may be satisfied immediately, they can benefit of VM placement strategies that support queues, priorities, and advance reservations [55].

Additionally, leases may be negotiated and renegotiated, allowing provider and consumer to modify a lease or present counter proposals until an agreement is reached. This feature is illustrated by the case in which an AR request for a given slot cannot be satisfied, but the provider can offer a distinct slot that is still satisfactory to the user. This problem has been addressed in OpenPEX, which incorporates a bilateral negotiation protocol that allows users and providers to come to an alternative agreement by exchanging offers and counter offers [56].

High Availability and Data Recovery. The high availability (HA) feature of VI managers aims at minimizing application downtime and preventing business disruption. A few VI managers accomplish this by providing a failover mechanism, which detects failure of both physical and virtual servers and restarts VMs on healthy physical servers. This style of HA protects from host, but not VM, failures [57, 58].

For mission critical applications, when a failover solution involving restarting VMs does not suffice, additional levels of fault tolerance that rely on redundancy of VMs are implemented. In this style, redundant and synchronized VMs (running or in standby) are kept in a secondary physical server. The HA solution monitors failures of system components such as servers, VMs, disks, and network and ensures that a duplicate VM serves the application in case of failures [58].

Data backup in clouds should take into account the high data volume involved in VM management. Frequent backup of a large number of VMs, each one with multiple virtual disks attached, should be done with minimal interference in the systems performance. In this sense, some VI managers offer data protection mechanisms that perform incremental backups of VM images. The backup workload is often assigned to proxies, thus offloading production server and reducing network overhead [59].

1.5.2 Case Studies

In this section, we describe the main features of the most popular VI managers available. Only the most prominent and distinguishing features of each tool are discussed in detail. A detailed side-by-side feature comparison of VI managers is presented in Table 1.1.

Apache VCL. The Virtual Computing Lab [60, 61] project has been incepted in 2004 by researchers at the North Carolina State University as a way to provide customized environments to computer lab users. The software components that support NCSU's initiative have been released as open-source and incorporated by the Apache Foundation.

Since its inception, the main objective of VCL has been providing desktop (virtual lab) and HPC computing environments anytime, in a flexible cost-effective way and with minimal intervention of IT staff. In this sense, VCL was one of the first projects to create a tool with features such as: self-service Web portal, to reduce administrative burden; advance reservation of capacity, to provide resources during classes; and deployment of customized machine images on multiple computers, to provide clusters on demand.

In summary, Apache VCL provides the following features: (i) multi-platform controller, based on Apache/PHP; (ii) Web portal and XML-RPC interfaces; (iii) support for VMware hypervisors (ESX, ESXi, and Server); (iv) virtual networks; (v) virtual clusters; and (vi) advance reservation of capacity.

AppLogic. AppLogic [62] is a commercial VI manager, the flagship product of 3tera Inc. from California, USA. The company has labeled this product as a Grid Operating System.

AppLogic provides a fabric to manage clusters of virtualized servers, focusing on managing multi-tier Web applications. It views an entire application as a collection of components that must be managed as a single entity. Several components such as firewalls, load balancers, Web servers, application servers, and database servers can be set up and linked together. Whenever the application is started, the system manufactures and assembles the virtual infrastructure required to run it. Once the application is stopped, AppLogic tears down the infrastructure built for it [63].

AppLogic offers dynamic appliances to add functionality such as Disaster Recovery and Power optimization to applications [62]. The key differential of this approach is that additional functionalities are implemented as another pluggable appliance instead of being added as a core functionality of the VI manager.

In summary, 3tera AppLogic provides the following features: Linux-based controller; CLI and GUI interfaces; Xen backend; Global Volume Store (GVS) storage virtualization; virtual networks; virtual clusters; dynamic resource allocation; high availability; and data protection.

TABLE 1.1. Feature Comparison of Virtual Infrastructure Managers

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

Citrix Essentials. The Citrix Essentials suite is one the most feature complete VI management software available, focusing on management and automation of data centers. It is essentially a hypervisor-agnostic solution, currently supporting Citrix XenServer and Microsoft Hyper-V [64].

By providing several access interfaces, it facilitates both human and programmatic interaction with the controller. Automation of tasks is also aided by a workflow orchestration mechanism.

In summary, Citrix Essentials provides the following features: Windows-based controller; GUI, CLI, Web portal, and XML-RPC interfaces; support for XenServer and Hyper-V hypervisors; Citrix Storage Link storage virtualization; virtual networks; dynamic resource allocation; three-level high availability (i.e., recovery by VM restart, recovery by activating paused duplicate VM, and running duplicate VM continuously) [58]; data protection with Citrix Consolidated Backup.

Enomaly ECP. The Enomaly Elastic Computing Platform, in its most complete edition, offers most features a service provider needs to build an IaaS cloud.

Most notably, ECP Service Provider Edition offers a Web-based customer dashboard that allows users to fully control the life cycle of VMs. Usage accounting is performed in real time and can be viewed by users. Similar to the functionality of virtual appliance marketplaces, ECP allows providers and users to package and exchange applications.

In summary, Enomaly ECP provides the following features: Linux-based controller; Web portal and Web services (REST) interfaces; Xen back-end; interface to the Amazon EC2 public cloud; virtual networks; virtual clusters (ElasticValet).

Eucalyptus. The Eucalyptus [39] framework was one of the first open-source projects to focus on building IaaS clouds. It has been developed with the intent of providing an open-source implementation nearly identical in functionality to Amazon Web Services APIs. Therefore, users can interact with a Eucalyptus cloud using the same tools they use to access Amazon EC2. It also distinguishes itself from other tools because it provides a storage cloud API—emulating the Amazon S3 API—for storing general user data and VM images.

In summary, Eucalyptus provides the following features: Linux-based controller with administration Web portal; EC2-compatible (SOAP, Query) and S3-compatible (SOAP, REST) CLI and Web portal interfaces; Xen, KVM, and VMWare backends; Amazon EBS-compatible virtual storage devices; interface to the Amazon EC2 public cloud; virtual networks.

Nimbus3. The Nimbus toolkit [20] is built on top of the Globus framework. Nimbus provides most features in common with other open-source VI managers, such as an EC2-compatible front-end API, support to Xen, and a backend interface to Amazon EC2. However, it distinguishes from others by

providing a Globus Web Services Resource Framework (WSRF) interface. It also provides a backend service, named Pilot, which spawns VMs on clusters managed by a local resource manager (LRM) such as PBS and SGE.

Nimbus' core was engineered around the Spring framework to be easily extensible, thus allowing several internal components to be replaced and also eases the integration with other systems.

In summary, Nimbus provides the following features: Linux-based controller; EC2-compatible (SOAP) and WSRF interfaces; Xen and KVM backend and a Pilot program to spawn VMs through an LRM; interface to the Amazon EC2 public cloud; virtual networks; one-click virtual clusters.

OpenNebula. OpenNebula is one of the most feature-rich open-source VI managers. It was initially conceived to manage local virtual infrastructure, but has also included remote interfaces that make it viable to build public clouds. Altogether, four programming APIs are available: XML-RPC and libvirt [47] for local interaction; a subset of EC2 (Query) APIs and the OpenNebula Cloud API (OCA) for public access [7, 65].

Its architecture is modular, encompassing several specialized pluggable components. The *Core* module orchestrates physical servers and their hypervisors, storage nodes, and network fabric. Management operations are performed through pluggable *Drivers*, which interact with APIs of hypervisors, storage and network technologies, and public clouds. The *Scheduler* module, which is in charge of assigning pending VM requests to physical hosts, offers dynamic resource allocation features. Administrators can choose between different scheduling objectives such as packing VMs in fewer hosts or keeping the load balanced. Via integration with the Haizea lease scheduler [66], OpenNebula also supports advance reservation of capacity and queuing of best-effort leases [7].

In summary, OpenNebula provides the following features: Linux-based controller; CLI, XML-RPC, EC2-compatible Query and OCA interfaces; Xen, KVM, and VMware backend; interface to public clouds (Amazon EC2, ElasticHosts); virtual networks; dynamic resource allocation; advance reservation of capacity.

OpenPEX. OpenPEX (Open Provisioning and EXecution Environment) was constructed around the notion of using advance reservations as the primary method for allocating VM instances. It distinguishes from other VI managers by its leases negotiation mechanism, which incorporates a bilateral negotiation protocol that allows users and providers to come to an agreement by exchanging offers and counter offers when their original requests cannot be satisfied.

In summary, OpenPEX provides the following features: multi-platform (Java) controller; Web portal and Web services (REST) interfaces; Citrix XenServer backend; advance reservation of capacity with negotiation [56].

oVirt. oVirt is an open-source VI manager, sponsored by Red Hat's Emergent Technology group. It provides most of the basic features of other VI managers,

including support for managing physical server pools, storage pools, user accounts, and VMs. All features are accessible through a Web interface [67].

The oVirt admin node, which is also a VM, provides a Web server, secure authentication services based on freeIPA, and provisioning services to manage VM image and their transfer to the managed nodes. Each managed node libvirt, which interfaces with the hypervisor.

In summary, oVirt provides the following features: Fedora Linux-based controller packaged as a virtual appliance; Web portal interface; KVM backend.

Platform ISF. Infrastructure Sharing Facility (ISF) is the VI manager offering from Platform Computing [68]. The company, mainly through its LSF family of products, has been serving the HPC market for several years.

ISF's architecture is divided into three layers. The top most *Service Delivery* layer includes the user interfaces (i.e., self-service portal and APIs); the *Allocation Engine* provides reservation and allocation policies; and the bottom layer—*Resource Integrations*—provides adapters to interact with hypervisors, provisioning tools, and other systems (i.e., external public clouds). The Allocation Engine also provides policies to address several objectives, such as minimizing energy consumption, reducing impact of failures, and maximizing application performance [44].

ISF is built upon Platform's VM Orchestrator, which, as a standalone product, aims at speeding up delivery of VMs to end users. It also provides high availability by restarting VMs when hosts fail and duplicating the VM that hosts the VMO controller [69].

In summary, ISF provides the following features: Linux-based controller packaged as a virtual appliance; Web portal interface; dynamic resource allocation; advance reservation of capacity; high availability.

VMWare vSphere and vCloud. vSphere is VMware's suite of tools aimed at transforming IT infrastructures into private clouds [36, 43]. It distinguishes from other VI managers as one of the most feature-rich, due to the company's several offerings in all levels the architecture.

In the vSphere architecture, servers run on the ESXi platform. A separate server runs vCenter Server, which centralizes control over the entire virtual infrastructure. Through the vSphere Client software, administrators connect to vCenter Server to perform various tasks.

The Distributed Resource Scheduler (DRS) makes allocation decisions based on predefined rules and policies. It continuously monitors the amount of resources available to VMs and, if necessary, makes allocation changes to meet VM requirements. In the storage virtualization realm, vStorage VMFS is a cluster file system to provide aggregate several disks in a single volume. VMFS is especially optimized to store VM images and virtual disks. It supports storage equipment that use Fibre Channel or iSCSI SAN.

In its basic setup, vSphere is essentially a private administration suite. Self-service VM provisioning to end users is provided via the vCloud API, which

interfaces with vCenter Server. In this configuration, vSphere can be used by service providers to build public clouds. In terms of interfacing with public clouds, vSphere interfaces with the vCloud API, thus enabling cloud-bursting into external clouds.

In summary, vSphere provides the following features: Windows-based controller (vCenter Server); CLI, GUI, Web portal, and Web services interfaces; VMware ESX, ESXi backend; VMware vStorage VMFS storage virtualization; interface to external clouds (VMware vCloud partners); virtual networks (VMWare Distributed Switch); dynamic resource allocation (VMware DRM); high availability; data protection (VMWare Consolidated Backup).

1.6 INFRASTRUCTURE AS A SERVICE PROVIDERS

Public Infrastructure as a Service providers commonly offer virtual servers containing one or more CPUs, running several choices of operating systems and a customized software stack. In addition, storage space and communication facilities are often provided.

1.6.1 Features

In spite of being based on a common set of features, IaaS offerings can be distinguished by the availability of specialized features that influence the cost–benefit ratio to be experienced by user applications when moved to the cloud. The most relevant features are: (i) geographic distribution of data centers; (ii) variety of user interfaces and APIs to access the system; (iii) specialized components and services that aid particular applications (e.g., load-balancers, firewalls); (iv) choice of virtualization platform and operating systems; and (v) different billing methods and period (e.g., prepaid vs. post-paid, hourly vs. monthly).

Geographic Presence. To improve availability and responsiveness, a provider of worldwide services would typically build several data centers distributed around the world. For example, Amazon Web Services presents the concept of "availability zones" and "regions" for its EC2 service. Availability zones are "distinct locations that are engineered to be insulated from failures in other availability zones and provide inexpensive, low-latency network connectivity to other availability zones in the same region." Regions, in turn, "are geographically dispersed and will be in separate geographic areas or countries [70]."

User Interfaces and Access to Servers. Ideally, a public IaaS provider must provide multiple access means to its cloud, thus catering for various users and their preferences. Different types of user interfaces (UI) provide different levels of abstraction, the most common being graphical user interfaces (GUI), command-line tools (CLI), and Web service (WS) APIs.

GUIs are preferred by end users who need to launch, customize, and monitor a few virtual servers and do not necessary need to repeat the process several times. On the other hand, CLIs offer more flexibility and the possibility of automating repetitive tasks via scripts (e.g., start and shutdown a number of virtual servers at regular intervals). WS APIs offer programmatic access to a cloud using standard HTTP requests, thus allowing complex services to be built on top of IaaS clouds.

Advance Reservation of Capacity. Advance reservations allow users to request for an IaaS provider to reserve resources for a specific time frame in the future, thus ensuring that cloud resources will be available at that time. However, most clouds only support best-effort requests; that is, users requests are server whenever resources are available [54].

Amazon Reserved Instances is a form of advance reservation of capacity, allowing users to pay a fixed amount of money in advance to guarantee resource availability at anytime during an agreed period and then paying a discounted hourly rate when resources are in use. However, only long periods of 1 to 3 years are offered; therefore, users cannot express their reservations in finer granularities—for example, hours or days.

Automatic Scaling and Load Balancing. As mentioned earlier in this chapter, elasticity is a key characteristic of the cloud computing model. Applications often need to scale up and down to meet varying load conditions. Automatic scaling is a highly desirable feature of IaaS clouds. It allow users to set conditions for when they want their applications to scale up and down, based on application-specific metrics such as transactions per second, number of simultaneous users, request latency, and so forth.

When the number of virtual servers is increased by automatic scaling, incoming traffic must be automatically distributed among the available servers. This activity enables applications to promptly respond to traffic increase while also achieving greater fault tolerance.

Service-Level Agreement. Service-level agreements (SLAs) are offered by IaaS providers to express their commitment to delivery of a certain QoS. To customers it serves as a warranty. An SLA usually include availability and performance guarantees. Additionally, metrics must be agreed upon by all parties as well as penalties for violating these expectations.

Most IaaS providers focus their SLA terms on availability guarantees, specifying the minimum percentage of time the system will be available during a certain period. For instance, Amazon EC2 states that "if the annual uptime Percentage for a customer drops below 99.95% for the service year, that customer is eligible to receive a service credit equal to 10% of their bill.[3]"

[3] http://aws.amazon.com/ec2-sla

Hypervisor and Operating System Choice. Traditionally, IaaS offerings have been based on heavily customized open-source Xen deployments. IaaS providers needed expertise in Linux, networking, virtualization, metering, resource management, and many other low-level aspects to successfully deploy and maintain their cloud offerings. More recently, there has been an emergence of turnkey IaaS platforms such as VMWare vCloud and Citrix Cloud Center (C3) which have lowered the barrier of entry for IaaS competitors, leading to a rapid expansion in the IaaS marketplace.

1.6.2 Case Studies

In this section, we describe the main features of the most popular public IaaS clouds. Only the most prominent and distinguishing features of each one are discussed in detail. A detailed side-by-side feature comparison of IaaS offerings is presented in Table 1.2.

Amazon Web Services. Amazon WS[4] (AWS) is one of the major players in the cloud computing market. It pioneered the introduction of IaaS clouds in 2006. It offers a variety cloud services, most notably: S3 (storage), EC2 (virtual servers), Cloudfront (content delivery), Cloudfront Streaming (video streaming), SimpleDB (structured datastore), RDS (Relational Database), SQS (reliable messaging), and Elastic MapReduce (data processing).

The Elastic Compute Cloud (EC2) offers Xen-based virtual servers (instances) that can be instantiated from Amazon Machine Images (AMIs). Instances are available in a variety of sizes, operating systems, architectures, and price. CPU capacity of instances is measured in Amazon Compute Units and, although fixed for each instance, vary among instance types from 1 (small instance) to 20 (high CPU instance). Each instance provides a certain amount of nonpersistent disk space; a persistence disk service (Elastic Block Storage) allows attaching virtual disks to instances with space up to 1TB.

Elasticity can be achieved by combining the CloudWatch, Auto Scaling, and Elastic Load Balancing features, which allow the number of instances to scale up and down automatically based on a set of customizable rules, and traffic to be distributed across available instances. Fixed IP address (Elastic IPs) are not available by default, but can be obtained at an additional cost.

In summary, Amazon EC2 provides the following features: multiple data centers available in the United States (East and West) and Europe; CLI, Web services (SOAP and Query), Web-based console user interfaces; access to instance mainly via SSH (Linux) and Remote Desktop (Windows); advanced reservation of capacity (aka reserved instances) that guarantees availability for periods of 1 and 3 years; 99.5% availability SLA; per hour pricing; Linux and Windows operating systems; automatic scaling; load balancing.

[4] http://aws.amazon.com

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

	Geographic Presence	Client UI API Language Bindings	Primary Access to Server	Advance Reservation of Capacity	SLA Uptime	Smallest Billing Unit	Hypervisor	Guest Operating Systems	Automated Horizontal Scaling	Load Balancing	Runtime Server Resizing/ Vertical Scaling	Instance Hardware Capacity		
												Processor	Memory	Storage
Amazon EC2	US East, Europe	CLI, WS, Portal	SSH (Linux), Remote Desktop (Windows)	Amazon reserved instances (Available in 1 or 3 years terms, starting from reservation time)	99.95%	Hour	Xen	Linux, Windows	Available with Amazon CloudWatch	Elastic Load Balancing	No	1–20 EC2 compute units	1.7–15 GB	160–1690 GB 1 GB–1 TB (per EBS volume)
Flexiscale	UK	Web Console	SSH	No	100%	Hour	Xen	Linux, Windows	No	Zeus software loadbalancing	Processors, memory (requires reboot)	1–4 CPUs	0.5–16 GB	20–270 GB
GoGrid		REST, Java, PHP, Python, Ruby	SSH	No	100%	Hour	Xen	Linux, Windows	No	Hardware (F5)	No	1–6 CPUs	0.5–8 GB	30–480 GB
Joyent Cloud	US (Emeryville, CA; San Diego, CA; Andover, MA; Dallas, TX)		SSH, VirtualMin (Web-based system administration)	No	100%	Month	OS Level (Solaris Containers)	OpenSolaris	No	Both hardware (F5 networks) and software (Zeus)	Automatic CPU bursting (up to 8 CPUs)	1/16–8 CPUs	0.25–32 GB	5–100 GB
Rackspace Cloud Servers	US (Dallas, TX)	Portal, REST, Python, PHP, Java, C#/.NET	SSH	No	100%	Hour	Xen	Linux	No	No	Memory, disk (requires reboot) Automatic CPU bursting (up to 100% of available CPU power of physical host)	Quad-core CPU (CPU power is weighed proportionally to memory size)	0.25–16 GB	10–620 GB

Flexiscale. Flexiscale is a UK-based provider offering services similar in nature to Amazon Web Services. However, its virtual servers offer some distinct features, most notably: persistent storage by default, fixed IP addresses, dedicated VLAN, a wider range of server sizes, and runtime adjustment of CPU capacity (aka CPU bursting/vertical scaling). Similar to the clouds, this service is also priced by the hour.

In summary, the Flexiscale cloud provides the following features: available in UK; Web services (SOAP), Web-based user interfaces; access to virtual server mainly via SSH (Linux) and Remote Desktop (Windows); 100% availability SLA with automatic recovery of VMs in case of hardware failure; per hour pricing; Linux and Windows operating systems; automatic scaling (horizontal/vertical).

Joyent. Joyent's Public Cloud offers servers based on Solaris containers virtualization technology. These servers, dubbed accelerators, allow deploying various specialized software-stack based on a customized version of OpenSolaris operating system, which include by default a Web-based configuration tool and several pre-installed software, such as Apache, MySQL, PHP, Ruby on Rails, and Java. Software load balancing is available as an accelerator in addition to hardware load balancers.

A notable feature of Joyent's virtual servers is automatic vertical scaling of CPU cores, which means a virtual server can make use of additional CPUs automatically up to the maximum number of cores available in the physical host.

In summary, the Joyent public cloud offers the following features: multiple geographic locations in the United States; Web-based user interface; access to virtual server via SSH and Web-based administration tool; 100% availability SLA; per month pricing; OS-level virtualization Solaris containers; OpenSolaris operating systems; automatic scaling (vertical).

GoGrid. GoGrid, like many other IaaS providers, allows its customers to utilize a range of pre-made Windows and Linux images, in a range of fixed instance sizes. GoGrid also offers "value-added" stacks on top for applications such as high-volume Web serving, e-Commerce, and database stores.

It offers some notable features, such as a "hybrid hosting" facility, which combines traditional dedicated hosts with auto-scaling cloud server infrastructure. In this approach, users can take advantage of dedicated hosting (which may be required due to specific performance, security or legal compliance reasons) and combine it with on-demand cloud infrastructure as appropriate, taking the benefits of each style of computing.

As part of its core IaaS offerings, GoGrid also provides free hardware load balancing, auto-scaling capabilities, and persistent storage, features that typically add an additional cost for most other IaaS providers.

Rackspace Cloud Servers. Rackspace Cloud Servers is an IaaS solution that provides fixed size instances in the cloud. Cloud Servers offers a range of Linux-based pre-made images. A user can request different-sized images, where the size is measured by requested RAM, not CPU.

Like GoGrid, Cloud Servers also offers hybrid approach where dedicated and cloud server infrastructures can be combined to take the best aspects of both styles of hosting as required. Cloud Servers, as part of its default offering, enables fixed (static) IP addresses, persistent storage, and load balancing (via A-DNS) at no additional cost.

1.7 PLATFORM AS A SERVICE PROVIDERS

Public Platform as a Service providers commonly offer a development and deployment environment that allow users to create and run their applications with little or no concern to low-level details of the platform. In addition, specific programming languages and frameworks are made available in the platform, as well as other services such as persistent data storage and in-memory caches.

1.7.1 Features

Programming Models, Languages, and Frameworks. Programming models made available by IaaS providers define how users can express their applications using higher levels of abstraction and efficiently run them on the cloud platform. Each model aims at efficiently solving a particular problem. In the cloud computing domain, the most common activities that require specialized models are: processing of large dataset in clusters of computers (MapReduce model), development of request-based Web services and applications; definition and orchestration of business processes in the form of workflows (Workflow model); and high-performance distributed execution of various computational tasks.

For user convenience, PaaS providers usually support multiple programming languages. Most commonly used languages in platforms include Python and Java (e.g., Google AppEngine), .NET languages (e.g., Microsoft Azure), and Ruby (e.g., Heroku). Force.com has devised its own programming language (Apex) and an Excel-like query language, which provide higher levels of abstraction to key platform functionalities.

A variety of software frameworks are usually made available to PaaS developers, depending on application focus. Providers that focus on Web and enterprise application hosting offer popular frameworks such as Ruby on Rails, Spring, Java EE, and .NET.

Persistence Options. A persistence layer is essential to allow applications to record their state and recover it in case of crashes, as well as to store user data.

Traditionally, Web and enterprise application developers have chosen relational databases as the preferred persistence method. These databases offer fast and reliable structured data storage and transaction processing, but may lack scalability to handle several petabytes of data stored in commodity computers [71].

In the cloud computing domain, distributed storage technologies have emerged, which seek to be robust and highly scalable, at the expense of relational structure and convenient query languages. For example, Amazon SimpleDB and Google AppEngine datastore offer schema-less, automatically indexed database services [70]. Data queries can be performed only on individual tables; that is, join operations are unsupported for the sake of scalability.

1.7.2 Case Studies

In this section, we describe the main features of some Platform as Service (PaaS) offerings. A more detailed side-by-side feature comparison of VI managers is presented in Table 1.3.

Aneka. Aneka [72] is a .NET-based service-oriented resource management and development platform. Each server in an Aneka deployment (dubbed Aneka cloud node) hosts the Aneka container, which provides the base infrastructure that consists of services for persistence, security (authorization, authentication and auditing), and communication (message handling and dispatching). Cloud nodes can be either physical server, virtual machines (XenServer and VMware are supported), and instances rented from Amazon EC2.

The Aneka container can also host any number of optional services that can be added by developers to augment the capabilities of an Aneka Cloud node, thus providing a single, extensible framework for orchestrating various application models.

Several programming models are supported by such task models to enable execution of legacy HPC applications and MapReduce, which enables a variety of data-mining and search applications.

Users request resources via a client to a reservation services manager of the Aneka master node, which manages all cloud nodes and contains scheduling service to distribute request to cloud nodes.

App Engine. Google App Engine lets you run your Python and Java Web applications on elastic infrastructure supplied by Google. App Engine allows your applications to scale dynamically as your traffic and data storage requirements increase or decrease. It gives developers a choice between a Python stack and Java. The App Engine serving architecture is notable in that it allows real-time auto-scaling without virtualization for many common types of Web applications. However, such auto-scaling is dependent on the

TABLE 1.3. Feature Comparison of Platform-as-a-Service Cloud Offerings

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

application developer using a limited subset of the native APIs on each platform, and in some instances you need to use specific Google APIs such as URLFetch, Datastore, and memcache in place of certain native API calls. For example, a deployed App Engine application cannot write to the file system directly (you must use the Google Datastore) or open a socket or access another host directly (you must use Google URL fetch service). A Java application cannot create a new Thread either.

Microsoft Azure. Microsoft Azure Cloud Services offers developers a hosted .NET Stack (C#, VB.Net, ASP.NET). In addition, a Java & Ruby SDK for .NET Services is also available. The Azure system consists of a number of elements. The Windows Azure Fabric Controller provides auto-scaling and reliability, and it manages memory resources and load balancing. The .NET Service Bus registers and connects applications together. The .NET Access Control identity providers include enterprise directories and Windows LiveID. Finally, the .NET Workflow allows construction and execution of workflow instances.

Force.com. In conjunction with the Salesforce.com service, the Force.com PaaS allows developers to create add-on functionality that integrates into main Salesforce CRM SaaS application.

Force.com offers developers two approaches to create applications that can be deployed on its SaaS plaform: a hosted Apex or Visualforce application. Apex is a proprietary Java-like language that can be used to create Salesforce applications. Visualforce is an XML-like syntax for building UIs in HTML, AJAX, or Flex to overlay over the Salesforce hosted CRM system. An application store called AppExchange is also provided, which offers a paid & free application directory.

Heroku. Heroku is a platform for instant deployment of Ruby on Rails Web applications. In the Heroku system, servers are invisibly managed by the platform and are never exposed to users. Applications are automatically dispersed across different CPU cores and servers, maximizing performance and minimizing contention. Heroku has an advanced logic layer than can automatically route around failures, ensuring seamless and uninterrupted service at all times.

1.8 CHALLENGES AND RISKS

Despite the initial success and popularity of the cloud computing paradigm and the extensive availability of providers and tools, a significant number of challenges and risks are inherent to this new model of computing. Providers, developers, and end users must consider these challenges and risks to take good advantage of cloud computing. Issues to be faced include user privacy, data

security, data lock-in, availability of service, disaster recovery, performance, scalability, energy-efficiency, and programmability.

1.8.1 Security, Privacy, and Trust

Ambrust et al. [5] cite information security as a main issue: "current cloud offerings are essentially public . . . exposing the system to more attacks." For this reason there are potentially additional challenges to make cloud computing environments as secure as in-house IT systems. At the same time, existing, well-understood technologies can be leveraged, such as data encryption, VLANs, and firewalls.

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. In this scenario, the trust toward providers is fundamental to ensure the desired level of privacy for applications hosted in the cloud [38].

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 determines the set of laws that can be applied to the management of data. For example, specific cryptography techniques could not be used because they are not allowed in some countries. Similarly, country laws can impose that sensitive data, such as patient health records, are to be stored within national borders.

1.8.2 Data Lock-In and Standardization

A major concern of cloud computing users is about having their data locked-in by a certain provider. Users may want to move data and applications out from a provider that does not meet their requirements. However, in their current form, cloud computing infrastructures and platforms do not employ standard methods of storing user data and applications. Consequently, they do not interoperate and user data are not portable.

The answer to this concern is standardization. In this direction, there are efforts to create open standards for cloud computing.

The Cloud Computing Interoperability Forum (CCIF) was formed by organizations such as Intel, Sun, and Cisco in order to "enable a global cloud computing ecosystem whereby organizations are able to seamlessly work together for the purposes for wider industry adoption of cloud computing technology." The development of the Unified Cloud Interface (UCI) by CCIF aims at creating a standard programmatic point of access to an entire cloud infrastructure.

In the hardware virtualization sphere, the Open Virtual Format (OVF) aims at facilitating packing and distribution of software to be run on VMs so that virtual appliances can be made portable—that is, seamlessly run on hypervisor of different vendors.

1.8.3 Availability, Fault-Tolerance, and Disaster Recovery

It is expected that users will have certain expectations about the service level to be provided once their applications are moved to the cloud. These expectations include availability of the service, its overall performance, and what measures are to be taken when something goes wrong in the system or its components. In summary, users seek for a warranty before they can comfortably move their business to the cloud.

SLAs, which include QoS requirements, must be ideally set up between customers and cloud computing providers to act as warranty. An SLA specifies the details of the service to be provided, including availability and performance guarantees. Additionally, metrics must be agreed upon by all parties, and penalties for violating the expectations must also be approved.

1.8.4 Resource Management and Energy-Efficiency

One important challenge faced by providers of cloud computing services is the efficient management of virtualized resource pools. Physical resources such as CPU cores, disk space, and network bandwidth must be sliced and shared among virtual machines running potentially heterogeneous workloads.

The multi-dimensional nature of virtual machines complicates the activity of finding a good mapping of VMs onto available physical hosts while maximizing user utility. Dimensions to be considered include: number of CPUs, amount of memory, size of virtual disks, and network bandwidth. Dynamic VM mapping policies may leverage the ability to suspend, migrate, and resume VMs as an easy way of preempting low-priority allocations in favor of higher-priority ones. Migration of VMs also brings additional challenges such as detecting when to initiate a migration, which VM to migrate, and where to migrate. In addition, policies may take advantage of live migration of virtual machines to relocate data center load without significantly disrupting running services. In this case, an additional concern is the trade-off between the negative impact of a live migration on the performance and stability of a service and the benefits to be achieved with that migration [73].

Another challenge concerns the outstanding amount of data to be managed in various VM management activities. Such data amount is a result of particular abilities of virtual machines, including the ability of traveling through space (i.e., migration) and time (i.e., checkpointing and rewinding) [74], operations that may be required in load balancing, backup, and recovery scenarios. In addition, dynamic provisioning of new VMs and replicating existing VMs require efficient mechanisms to make VM block storage devices (e.g., image files) quickly available at selected hosts.

Data centers consumer 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 CO_2 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.

1.9 SUMMARY

Cloud computing is a new computing paradigm that offers a huge amount of compute and storage resources to the masses. Individuals (e.g., scientists) and enterprises (e.g., startup companies) can have access to these resources by paying a small amount of money just for what is really needed.

This introductory chapter has surveyed many technologies that have led to the advent of cloud computing, concluding that this new paradigm has been a result of an evolution rather than a revolution.

In their various shapes and flavors, clouds aim at offering compute, storage, network, software, or a combination of those "as a service." Infrastructure-, Platform-, and Software-as-a-service are the three most common nomenclatures for the levels of abstraction of cloud computing services, ranging from "raw" virtual servers to elaborate hosted applications.

A great popularity and apparent success have been visible in this area. However, as discussed in this chapter, significant challenges and risks need to be tackled by industry and academia in order to guarantee the long-term success of cloud computing. Visible trends in this sphere include the emergence of standards; the creation of value-added services by augmenting, combining, and brokering existing compute, storage, and software services; and the availability of more providers in all levels, thus increasing competiveness and innovation. In this sense, numerous opportunities exist for practitioners seeking to create solutions for cloud computing.

REFERENCES

1. I. Foster, The grid: Computing without bounds, *Scientific American*, vol. 288, No. 4, (April 2003), pp. 78–85.
2. R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility, *Future Generation Computer Systems*, **25**:599–616, 2009.
3. L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, A break in the clouds: Towards a cloud definition, *SIGCOMM Computer Communications Review*, **39**:50–55, 2009.
4. McKinsey & Co., Clearing the Air on Cloud Computing, *Technical Report*, 2009.

5. M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, and R. Katz, Above the Clouds: A Berkeley View of Cloud Computing, *UC Berkeley Reliable Adaptive Distributed Systems Laboratory White Paper*, 2009.
6. P. Mell and T. Grance, The NIST Definition of Cloud Computing, National Institute of Standards and Technology, Information Technology Laboratory, *Technical Report Version 15*, 2009.
7. B. Sotomayor, R. S. Montero, I. M. Llorente, and I. Foster, Virtual infrastructure management in private and hybrid clouds, *IEEE Internet Computing*, **13**(5):14–22, September/October, 2009.
8. N. Carr, *The Big Switch: Rewiring the World*, from Edison to Google.W. W. Norton & Co., New York, 2008.
9. M. A. Rappa, The utility business model and the future of computing systems, *IBM Systems Journal*, **43**(1):32–42, 2004.
10. C. S. Yeo et al., Utility computing on global grids, Chapter 143, Hossein Bidgoli (ed.), *The Handbook of Computer Networks*, ISBN: 978-0-471-78461-6, John Wiley & Sons, New York, USA, 2007.
11. I. Foster and S. Tuecke, Describing the elephant: The different faces of IT as service, *ACM Queue*, **3**(6):26–29, 2005.
12. M. P. Papazoglou and W.-J. van den Heuvel, Service oriented architectures: Approaches, technologies and research issues, *The VLDB Journal*, **16**:389–415, 2007.
13. H. Kreger, Fulfilling the Web services promise, *Communications of the ACM*, **46**(6):29, 2003.
14. B. Blau, D. Neumann, C. Weinhardt, and S. Lamparter, Planning and pricing of service mashups, in *Proceedings of the 2008 10th IEEE Conference on E-Commerce Technology and the Fifth IEEE Conference on Enterprise Computing, E-Commerce and E-Services*, Crystal City, Washington, DC, 2008, pp.19–26.
15. C. Catlett, The philosophy of TeraGrid: Building an open, extensible, distributed TeraScale facility, in *Proceedings of 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid*, Berlin, Germany, 2002, p. 8.
16. F. Gagliardi, B. Jones, F. Grey, M. E. Begin, and M. Heikkurinen, Building an infrastructure for scientific grid computing: Status and goals of the EGEE project, *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, **363**(1833):1729, 2005.
17. J. Broberg, S. Venugopal, and R. Buyya, Market-oriented Grid and utility computing: The state-of-the-art and future directions, *Journal of Grid Computing*, **6**:255–276, 2008.
18. I. Foster, Globus toolkit version 4: Software for service-oriented systems, *Journal of Computer Science and Technology*, **21**(513–520), 2006.
19. R. Buyya and S. Venugopal, *Market oriented computing and global Grids: An introduction*, in *Market Oriented Grid and Utility Computing*, R. Buyya and K. Bubendorfer (eds.), John Wiley & Sons, Hoboken, NJ, 2009, pp. 24–44.
20. K. Keahey, I. Foster, T. Freeman, and X. Zhang, Virtual workspaces: Achieving quality of service and quality of life in the grid, *Scientific Programming*, **13**(4):265–275, 2005.
21. R. P. Goldberg, Survey of virtual machine research, *IEEE Computer*, **7**(6):34–45, 1974.

22. R. Uhlig et al., Intel virtualization technology, *IEEE Computer*, **38**(5):48–56, 2005.
23. P. Barham et al., Xen and the art of virtualization, in *Proceedings of 19th ACM Symposium on Operation Systems Principles*, New York, 2003, pp. 164–177.
24. VMWare Inc., VMWare, *http://www.vmware.com*, 22/4/2010.
25. Xen.org Community, *http://www.xen.org*, 22/4/2010.
26. Citrix Systems Inc., XenServer, *http://www.citrix.com/XenServer*, 22/4/2010.
27. Oracle Corp., Oracle VM, *http://www.oracle.com/technology/products/vm*, 24/4/2010.
28. KVM Project, Kernel based virtual machine, *http://www.linux-kvm.org*, 22/4/2010.
29. A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori, KVM: The Linux virtual machine monitor, in *Proceedings of the Linux Symposium*, Ottawa, Canada, 2007, p. 225.
30. VMWare Inc., VMWare Virtual Appliance Marketplace, *http://www.vmware.com/appliances*, 22/4/2010.
31. Amazon Web Services Developer Community, Amazon Machine Images, *http://developer.amazonwebservices.com/connect/kbcategory.jspa?categoryID = 171*, 22/4/2010.
32. Distributed Management Task Force Inc, Open Virtualization Format, *Specification DSP0243 Version 1.0.0*, 2009.
33. J. Matthews, T. Garfinkel, C. Hoff, and J. Wheeler, Virtual machine contracts for datacenter and cloud computing environments, in *Proceedings of the 1st Workshop on Automated Control for Datacenters and Clouds*, 2009, pp. 25–30.
34. International Business Machines Corp., An architectural blueprint for autonomic computing, *White Paper Fourth Edition*, 2006.
35. M. C. Huebscher and J. A. McCann, A survey of autonomic computing—degrees, models, and applications, *ACM Computing Surveys*, **40**:1–28, 2008.
36. VMWare Inc., VMware vSphere, *http://www.vmware.com/products/vsphere/*, 22/4/2010.
37. L. Youseff, M. Butrico, and D. Da Silva, Toward a unified ontology of cloud computing, in *Proceedings of the 2008 Grid Computing Environments Workshop*, 2008, pp. 1–10.
38. R. Buyya, S. Pandey, and C. Vecchiola, Cloudbus toolkit for market-oriented cloud computing, in *Proceedings 1st International Conference on Cloud Computing (CloudCom 09)*, Beijing, 2009, pp. 3–27.
39. D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, The Eucalyptus open-source cloud-computing system, in *Proceedings of IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2009)*, Shanghai, China, pp. 124–131, University of California, Santa Barbara. (2009, Sep.) Eucalyptus [online]. http://open.eucalyptus.com.
40. Appistry Inc., Cloud Platforms vs. Cloud Infrastructure, *White Paper*, 2009.
41. B. Hayes, Cloud computing, *Communications of the ACM*, **51**:9–11, 2008.
42. P. T. Jaeger, J. Lin, J. M. Grimes, and S. N. Simmons, Where is the cloud? Geography, economics, environment, and jurisdiction in cloud computing, *First Monday*, **14**(4–5): 2009.
43. VMWare Inc., VMware vSphere, the First Cloud Operating, *White Paper*, 2009.
44. Platform Computing, Platform ISF Datasheet, *White Paper*, 2009.

45. M. D. de Assuncao, A. di Costanzo, and R. Buyya, Evaluating the cost–benefit of using cloud computing to extend the capacity of clusters, in *Proceedings of the 18th ACM International Symposium on High Performance Distributed Computing (HPDC 2009)*, Munich, Germany, 2009, pp. 141–150.

46. D. Amrhein, Websphere Journal, *http://websphere.sys-con.com/node/1029500*, 22/4/2010.

47. Libvirt: The Virtualization API, Terminology and Goals, *http://libvirt.org/goals.html*, 22/4/2010.

48. A. Singh, M. Korupolu, and D. Mohapatra, Server-storage virtualization: Integration and load balancing in data centers, in *Proceedings of the 2008 ACM/IEEE Conference on Supercomputing*, 2008, pp. 1–12.

49. R. Perlman, *Interconnections: Bridges, Routers, Switches, and Internetworking Protocols*, Addison-Wesley Longman, Boston, MA, 1999.

50. A. S. Tanenbaum, *Computer Networks*, Prentice-Hall, Upper Saddle River, NJ, 2002.

51. D. Gmach, J. Rolia, L. Cherkasova, and A. Kemper, Capacity management and demand prediction for next generation data centers, in *Proceedings of IEEE International Conference on Web Services*, 2007, pp. 43—50.

52. A. Verma, P. Ahuja, and A. Neogi, pMapper: Power and migration cost aware application placement in virtualized systems, in *Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware*, 2008, pp. 243—264.

53. K. Keahey and T. Freeman, Contextualization: Providing one-click virtual clusters, in *Proceedings of IEEE Fourth International Conference on eScience*, 2008, pp. 301—308.

54. B. Sotomayor, K. Keahey, and I. Foster, Combining batch execution and leasing using virtual machines, in *Proceedings of the 17th International Symposium on High Performance Distributed Computing*, 2008, pp. 87—96.

55. B. Sotomayor, R. Montero, I. M. Llorente, and I. Foster, Capacity leasing in cloud systems using the opennebula engine, *Cloud Computing and Applications*, 2008.

56. S. Venugopal, J. Broberg, and R. Buyya, OpenPEX: An open provisioning and EXecution system for virtual machines, in *Proceedings of the 17th International Conference on Advanced Computing and Communications (ADCOM 2009)*, Bengaluru, India, 2009.

57. VMWare Inc., VMware High Availability (HA), *http://www.vmware.com/products/high-availability/index.html*, 22/4/2010.

58. Citrix Systems Inc., The three levels of high availability—Balancing priorities and cost, *White Paper*, 2008.

59. VMWare Inc., VMWare vStorage APIs for Data Protection, *http://www.vmware.com/products/vstorage-apis-for-data-protection*, 22/4/2010.

60. H. E. Schaffer et al., NCSUs Virtual Computing Lab: A cloud computing solution, *Computer*, **42**:94–97, 2009.

61. North Carolina State University, Virtual Computing Lab (VCL), *http://vcl.ncsu.edu*, 22/4/2010.

62. 3tera Inc., AppLogic—Grid Operating System for Web Applications, *http://www.3tera.com/AppLogic*, 22/4/2010.

63. 3Tera Inc., The AppLogic Grid Operating System, *White Paper*, 2006.

64. Citrix Systems Inc., Citrix essentials for Hyper-V, *http://www.citrix.com/ehv*, 22/4/2010.
65. Distributed Systems Architecture Group, OpenNebula: The open source toolkit for cloud computing, *http://www.opennebula.org*, 22/4/2010.
66. University of Chicago, Haizea—An open source VM-based lease manager, *http://haizea.cs.uchicago.edu*, 22/4/2010.
67. Red Hat's Emerging Technology group, oVirt, *http://ovirt.org*, 22/4/2010.
68. Platform Computing Corporation, Platform ISF. *http://www.platform.com/Products/platform-isf*, 22/4/2010.
69. Platform Computing, Platform VM Orchestrator, *http://www.platform.com/Products/platform-vm-orchestrator*, 22/4/2010.
70. Amazon Inc., Amazon Web Services, *http://www.amazon.com*, 22/4/2010.
71. F. Chang et al., Bigtable: A distributed storage system for structured data, in *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation (OSDI'06)*, 2006, pp. 205–218.
72. C. Vecchiola, X. Chu, and R. Buyya, Aneka: A software platform for .NET-based cloud computing, in *High Speed and Large Scale Scientific Computing*, W. Gentzsch, L. Grandinetti, and G. Joubert (eds.), IOS Press, Amsterdam, Netherlands, 2009, pp. 267–295.
73. W. Voorsluys, J. Broberg, S. Venugopal, and R. Buyya, Cost of virtual machine live migration in clouds: A performance evaluation, in *Proceedings 1st International Conference on Cloud Computing*, Beijing, 2009, pp. 254–265.
74. D. T. Meyer et al., Parallax: Virtual disks for virtual machines, in *Proceedings of the 3rd ACM SIGOPS/EuroSys European Conference on Computer Systems*, 2008, pp. 41–54.