Cloud Computing Uncovered: A Research Landscape

MOHAMMAD HAMDAQA AND
LADAN TAHVILDARI

Software Technologies Applied Research (STAR) Group, University of Waterloo, Waterloo, Ontario, Canada

Abstract

The amalgamation of technology and service-based economy was the impetus of a new IT delivery model called "Cloud Computing." Unsurprisingly, the new model created a state of confusion; new concepts are mixed with old ones, and some old technologies are being reinvented. Today, many research areas and projects under the cloud umbrella need to be sorted and classified. Additionally, many gaps need to be categorized so that each research area can be tackled by its experts. It is important to learn from what already exists, and to be able to distinguish between what is new and what can be reused. Accordingly, this survey article presents a unified view of the Cloud Computing main concepts, characteristics, models, and architectures. Moreover, it presents a landscape of research in Cloud Computing by highlighting relevant disciplines and some diligent research projects. This helps identify the underlying research gaps and elaborates on the corresponding challenges.

ISSN: 0065-2458, http://dx.doi.org/10.1016/B978-0-12-396535-6.00002-8.

1. Introduction

While Cloud Computing is currently a hype, neither the concept nor the technology behind it is new. It is believed that the concept of Cloud Computing is the same as what John McCarthy, in the 1960s, referred to as the ability to provide and organize computation as a "utility." The main characteristics of the Cloud Computing were also discussed by Parkhill [1]. On the other hand, the term *cloud* and its graphical symbol have been used for decades in computer network literature, first to refer to the large Asynchronous Transfer Mode (ATM) networks in the 1990s, and then to describe the Internet (a large number of distributed computers).

Even though the concept is not new, today there are hundreds if not thousands of attempts to define "Cloud Computing." For example, Vaquero et al. compared 22 different definitions in an attempt to provide a unified one [2]. Some of these definitions are general: "applications delivered as services over the Internet and the hardware and systems software in the datacenters that provide those services" [3]. Others are more specific: "A Cloud is a type of parallel and distributed system consisting of a collection of inter-connected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on Service Level Agreements (SLAs) established through negotiation between the service provider and consumers." The fact that Cloud Computing is not a pure technical term, as well as the large number of interdisciplinary technologies that participate in characterizing Cloud Computing, are the reasons behind all the different definitions.

The information technology industry has sought to standardize the definition of Cloud Computing. One of the first standardized definitions is the one by Forrester Research, Inc; in which they defined Cloud Computing as “A standardized IT capability (services, software, or infrastructure) delivered via Internet technologies in a pay-per-use, self-service way” [4]. Forrester’s definition focuses on the service models and business model of the cloud; however, it ignores the deployment models. This is because Forrester does not believe in private clouds, as a way to cut costs. According to a July 2010 Forrester Research paper entitled “You’re Not Ready for Internal Cloud” [5], building a private cloud datacenter that satisfies all the technological and legal requirements is a daunting task. The cost of which outweighs the benefits gained. The most recent and accepted standardized definition of Cloud Computing is the one by the National Institute of Standards and Technology (NIST) [6]:

> “Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models.”

The NIST definition is relatively technical, and covers all of the cloud service (IaaS, PaaS, SaaS) and deployment (public, private, hybrid, community) models, as shown in Fig. 1. The NIST definition is concise, accurate, and distinguishes between characteristics and enabling technologies (i.e., virtualization), and between main and derived characteristics (rapid elasticity vs. massive scalability); nevertheless, the definition ignores the business model of Cloud Computing, which is the main driver for moving to the cloud. However, it is worth mentioning that the “pay-per-use” model was not omitted from the definition out of ignorance, but because NIST tried to cover all of the cloud deployment models, including the “Private Cloud,” which not necessarily involves the “pay-per-use” practice. Instead of explaining the business model, NIST identified the main technological characteristics that can result in cost reduction, and add constraints (e.g., all services must be measurable) that provide all the requirements for cloud deployment models to adopt any billing or utility model (i.e., “pay-per-use”).

The rest of this chapter will be organized as follows: Section 2 explains the Cloud Computing principles and requirements. Section 3 investigates Cloud Computing reference models, architectures and frameworks. Section 4 clarifies the relationship between Cloud Computing and the Service-Oriented Architecture (SOA),

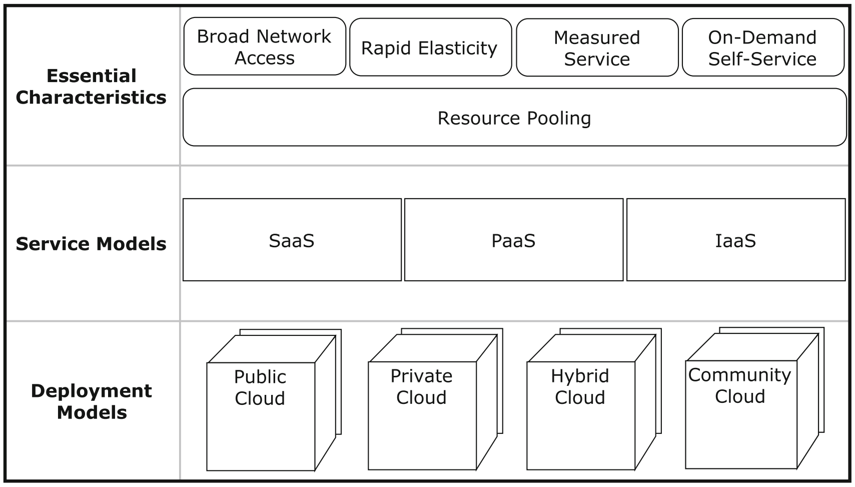


FIG. 1. NIST visual model of cloud computing definition.

grid computing, parallel computing, utility computing, autonomic computing, and virtualization. Finally, this chapter is concluded in Section 5.

2. Cloud Computing Principles and Requirements

The NIST definition of Cloud Computing reveals the main characteristics, delivery models, and service models of Cloud Computing. This section describes the foundation of Cloud Computing by listing and explaining these characteristics and models in more detail.

2.1 Cloud Computing Characteristics

The following are the five main characteristics of Cloud Computing that most people agree upon:

(a) *On-demand self-service:* Cloud services are on-demand; that is, service consumers can automatically request the service based on their needs, without human interaction with the service provider.

(b) *Easy to access standardized mechanisms:* NIST refers to this characteristic as broad network access; however, the term "global reach capability" is also used. The idea is that it should be possible to access cloud services through the network using standardized interfaces and access mechanisms. Having global reach capability does not mean that these services must always be accessible from the Internet, because this depends on the deployment model used. However, it should be possible to reach the service globally, when policies allow this.

(c) *Resource pooling and multi-tenancy:* In Cloud Computing, resources [i.e., storages, processors, memory, network bandwidth, and virtual machines (VMs)] are shared between multiple tenants, and assigned exclusively at run time to one consumer at a time. Assigning resources is done dynamically based on the consumers' needs. Sharing resources can help increase utilization, and hence significantly reduce the operation cost. Scheduling algorithms can be used to dynamically assign resources to different tenants based on the type of workload, fairness, locality, and many other factors [7, 8].

(d) *Rapid elasticity:* Elasticity is the ability to scale in and out by provisioning resources and releasing them, respectively. Cloud Computing should provide mechanisms to allow quick and automatic elasticity. The large pool of resources in cloud datacenters gives the illusion of infinite resources to the consumers, and elasticity provides the flexibility to provision these recourses on-demand.

(e) *Measured service:* Providing cloud metrology or mechanisms to measure service usage as well as to monitor the health of services is crucial in Cloud Computing. Measuring services enables optimizing resources and provides transparency for both consumers and providers, allowing them to better utilize the service. Measured services can help in building closed-loop cloud systems that are fully automated.

(f) *Auditability and certifiability:* Regulatory compliance requires enforcing rules and regulations. Services should provide logs and trails that allow the traceability of policies, so as to ensure that they are correctly enforced.

The list above (except point f) is based on the NIST definition. The list describes Cloud Computing based on what is currently available in the market, and represents the main characteristics of Cloud Computing in general. Many other characteristics can be added to this list in the future. For example, we added "auditability and certifiability" to the list above based on the current regulatory compliance requirements. On the other hand, comprehensive lists of characteristics can be made for each layer and each type of service provided in the cloud environment. For example, at the application level, a possible cloud service characteristic is that a service must be portable, pre-configured, or adaptable.

2.2 Cloud Computing Deployment Models

A Cloud Computing deployment model is a model that describes the environment where cloud applications and services can be installed, in order to be available to consumers. By the deployment environment, we mean the physical location, the infrastructure facilities, the platform constraints, as well as anything that can affect the access mechanisms of the deployed applications. There are four main Cloud Computing deployment models: public, private, hybrid, and community cloud:

(a) *Public cloud:* A public cloud or external cloud is an open model, in which the infrastructure facilities are provided by a third party (the cloud providers). The infrastructure and platform services are provided to the public based on the service level agreement between the provider and the consumer. This type of infrastructure resource sharing between multiple organizations or consumers, is referred to as the multi-tenancy model. Public cloud is the least expensive choice for application hosting. However, the lack of a trust model between the cloud providers and consumers is the main obstacle for this model.

(b) *Private cloud:* A private cloud or internal cloud is a datacenter owned by a cloud application provider, in which the infrastructure and platform are operated entirely by the application provider on premises. This eliminates the need for a trust model and provides more flexibility. Organizations can implement their own policies with regards to privacy, security, and access mechanisms. However, this option is expensive in terms of resources, and the manpower needed to manage the resources.

(c) *Hybrid cloud:* A hybrid cloud is a combination of a public and private cloud. A hybrid cloud is less expensive than a private cloud; it also eliminates the need for a trust model. However, having both public and private clouds working together requires interoperability and portability of both applications and data to allow communication between the models.

(d) *Community (cooperative) cloud:* A community cloud is similar to extranets, but with virtualization and on-demand capabilities. In a community cloud, a number of organizations, which usually share some common goals or belong to a specific community, build a shared cloud datacenter that can be used by all of the members. The goals are to alleviate deficiencies in the individual IT infrastructures, reduce the cost of administration, and lower the cost per unit [9]. The community can be created between a professional community (i.e., organizations with business relationship), a geographic community, or some other well-defined community group. Community cloud is based on the trust relation between all the members, which is driven by their mutual benefits [10,11]. As a result, this model is more trusted than the public cloud, and less expensive on participating members than having a private cloud. This model also provides more controllability over the shared infrastructure resources. However, a community cloud still needs to enforce strong security and privacy policies. Furthermore, regulatory compliance is a main obstacle facing community cloud adoption.

Table I shows a comparison between the different cloud deployment models, based on the initial cost of building the cloud datacenter or the capital expenses (CapEx) on the consumer, the operating expenses (OpEx) and maintenance cost of the datacenter, the size of the datacenter, controllability and flexibility, the level of trust, the location of the infrastructure, and who owns the infrastructure.

As shown in Table I, there is no initial cost associated with adopting public cloud by consumers [12]. Consumers need not worry about creating the cloud infrastructure. Instead, they can request the services and resources on-demand and pay just for what they use. Conversely, a private cloud requires a big initial capital investment in order to build the private datacenter [12]. Unlike the private model, the hybrid model builds a relatively small private datacenter for sensitive and important tasks and information, and uses the public cloud for other jobs. For this reason, the cost of adopting the hybrid cloud model is between that of public and private clouds. Finally, the community cloud model shares the cost of building the required datacenter with the cooperatives. For this reason, the initial cost can vary; the larger the community the smaller the share and the lower the cost.

Table I also shows that the operating cost (i.e., power consumption, man power expenses, rent, maintenance, upgrades, etc.) of public cloud is lower than the other models. This is due to the economies of scale, as well as the high level of automation and optimization in public cloud. Production costs drop significantly as the number of units produced increase[1] [13]. This allows public cloud providers to enjoy favorable prices for IT equipment and needed resources, since they purchase them in bulk. According to Jon Moore's blog, a private datacenter should have on average 10,000 servers to get an economically feasible marginal cost that is comparable to what current public cloud providers charge [14]. On the other hand, public providers tend to invest more in automation and optimization, which results in fewer administrative staff. For example, while the ratio between IT staff to servers is (1:100) [15] in traditional datacenters, this ratio goes to (1:1000) [15] and even (1:5000) in public cloud datacenters.

It is clear from Table I that consumers can have full control over a private cloud infrastructure, whereas in a public cloud, controllability is limited to tuning some

[1] This phenomenon is referred to as the experience curve effect and was first noticed by Bruce Henderson in 1960 at BCG (Boston Consulting Group).

TABLE I
A COMPARISON BETWEEN THE DIFFERENT CLOUD DEPLOYMENT MODELS.

Attribute	Public cloud	Private cloud	Hybrid cloud	Community cloud
Cost of building the datacenter on service consumer	No initial cost	High initial cost	Medium initial cost	Varies depends on the number of cooperatives
Operation and maintenance cost on the provider	Lowest cost with respect to the datacenter size	Highest cost with respect to the datacenter size	Weighted average, depending on the percentage of public and private parts	Similar to private clouds, but the cost divided on the participants
Size of the datacenter	~50,000 server	~50,000 server	Less than private cloud	~15,000 more than private cloud but much less than public cloud
Infrastructure controllability and flexibility	Limited configuration controllability	Full controllability (HW + SW)	Full controllability over the private part and limited for the public part	High controllability but limited by the community policies
Level of trust	Lowest trust	Highest	Medium trust	High trust
Infrastructure location	Off-premise	On-premise	Both on- and off-premise	Within the cooperative facility
Owner of the infrastructure	The laaS vendor	The customer	The laaS vendor owns the public part and the consumer owns the in-house part	Shared between the cooperatives

configuration parameters. On the other hand, while community cloud consumers can have access and control over the infrastructure, this controllability is bounded by the community policies and agreements.

The level of controllability and flexibility can also affect the level of trust. This explains why consumers trust the private cloud model more than the other models. However, it is important to note that the level of trust is not related to the actual security level. Public cloud providers tend to implement best practices and try to ensure security at every level of the security stack. However, the Cloud Computing paradigm introduces new security threats that did not exist in traditional datacenters, such as threats related to sharing resources through virtualization [16]. Most of these threats are equally applicable to both public and private models. Some of the cloud security myths assume that all clouds are created equally [17], while others assume that public and private cloud providers have the same experience and capabilities to implement security measures [18] for data protection, identity management, compliance, access control rules, and other security capabilities. If these assumptions are true, then public cloud is the least secure, while private cloud is the most secure.

Cloud deployment models differ based on the infrastructure's owner, location, or operators and their policies. One model does not fit all business types. The selection of a cloud deployment model depends on the consumers' needs and budget, and on whether they favor price reduction and control delegation over flexibility, control, and customization.

2.3 Cloud Computing Service Models (Cloud Service Hierarchical View)

This subsection presents and compares the Cloud Computing service models. Cloud service models are sometimes referred to as the cloud service hierarchical view [19], cloud service offerings, cloud service delivery models [20], or the cloud service layered architecture, in an analogy to the network layered architecture. Cloud service models try to classify "*anything*" providers offer *as a service* (XaaS), where X means an arbitrary service (e.g., infrastructure, software, storage). A cloud service model represents a layered high-level abstraction of the main classes of services provided by the Cloud Computing model, and how these layers are connected to each other. This separation between layers allows each cloud provider to focus on the core services they provide, while at the same time being able to reuse the services from the lower layers by following the set of standard communication mechanisms and protocols between the different layers. Layers differ based on the management scope covered by the provider [21], which means that a user in the upper layers cannot bypass

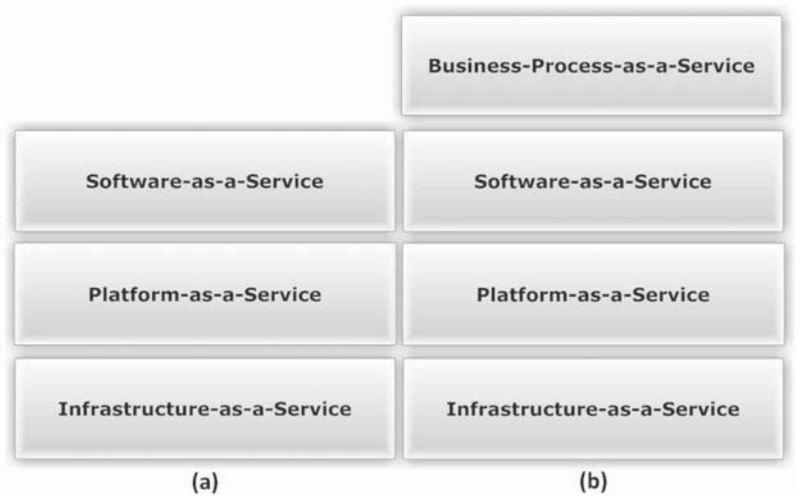


FIG. 2. (a) SPI service model vs. (b) IBM service model.

the interfaces provided by the layer beneath, so as to directly access the resources. This separation does not only help in service integration but also allows having a fully distributed, scalable, and fault tolerant architecture. By having different layers with different abstraction levels, cloud providers can have better manageability over the resources, as well as higher controllability and security.

As in the network layered architectures (i.e., OSI, TCP/IP), there are different cloud service models. These models vary based on the time they were proposed, relative to the maturity of the Cloud Computing paradigm at that time; and on the level of detail in these models, as represented in the number of model layers. However, the differences between service models do not contradict each other, instead these models are complementary [22].

This subsection will briefly discuss the main two service models: The NIST SPI model [6] (aka, the three service-layers model), and the IBM service model [21] (aka, the four service-layers model). Figure 2 shows a comparison between these two models.

2.3.1 The NIST SPI Model

The SPI model classifies the services provided by the cloud providers into three main categories (layers): *S*oftware services, *P*latform services, and *I*nfrastructure services. The SPI model is named after these categories, which are described below:

- *Software as a Service (SaaS):* A service is classified as a software service if it allows the consumer (end user) to access and use a provider software application that is owned (hosted), deployed, and managed by the provider. Consumers normally have limited control over the application, and are restricted in how they can use and interact with the application. The application is usually accessed via a thin client (i.e., Web browser), through which consumers can input data and get output [6]. Because most SaaS services are specific applications rather than being generic software services, SaaS is sometimes referred to as Application-as-a-Service. Examples of SaaS are content services such as video-on-demand (i.e., Netflix), email services (i.e., Gmail), and business applications such as customer relationship management applications (i.e., Salesforce).
- *Platform as a Service:* A service is classified as a platform service if it allows the service consumer (usually a SaaS provider, cloud developer, or administrator) to define, develop, configure, deploy, manage, and monitor cloud applications. While PaaS allows consumers to deploy and control applications and their hosting environment configurations, consumers do not have direct control over the underlying cloud infrastructure [6]. PaaS services abstract the communications with the lower-level infrastructure by providing easy to access and easy to use interfaces. Operating systems and application frameworks are part of the PaaS layer.
- *Infrastructure-as-a-Service:* A service is classified as an infrastructure service if it allows the service consumer (usually PaaS providers) to lease infrastructure capabilities based on demand. The infrastructure capabilities include processing, storage, network, or any other basic computing resources that can be used to deploy and run platforms (i.e., operating systems, management tools, development tools, and monitoring tools) and the applications developed on top of the platforms. Again, consumers are not given direct access to resources but have the ability to select and configure resources as required based on their needs [6]. IaaS is sometimes referred to as the virtualization layer, since it utilizes virtualization technologies to partition physical resources, so as to provide the consumers with a pool of storage and computing resources.

2.3.2 IBM Service Model

According to the IBM Cloud Computing Reference Architecture [21], a cloud service model consists of four service layers; from top to down, these are the Business-Process, Software, Platform, and Infrastructure-as-a-Service layer.

The last three layers are exactly the same as in the SPI model. In fact, IBM used the same definition of SaaS, PaaS, and IaaS as defined according to the NIST SPI model. The only difference between the IBM service model and the NIST SPI is in the Business-Process-as-a-Service (BPaaS) layer. Since this is the only difference, we will only explain BPaaS and why it was introduced:

- *Business-Process-as-a-Service:* A service is classified as a business process service if it allows the consumer (end user, business process manager, or designer) to design, manage, and integrate a set of transactional and collaborative activities based on the SaaS provided in the layer beneath, so as to accomplish a specific business organizational goal. Accordingly, IBM classifies any business process service—whether it focuses on technology and reuse (horizontal), or it is domain specific (vertical)—as BPaaS if and only if the service (1) represents a business process that is delivered through the Cloud Computing model based on its main characteristics, as defined in the NIST definition (i.e., multi-tenant service, self-service, elastically scaled, metered, and priced); (2) accessed through a Web-centric interface; and (3) utilizes Web-oriented cloud architectures. Similar to IaaS and PaaS services, a BPaaS provider provides the tools to access and utilize the resources in the BPaaS layer. Consumers do not need to access services in the underlying layers. "A BPaaS provider is responsible for the related business function(s)" [21]. Some examples of BPaaS include a process for employee benefit management; and IT-centric processes, such as a process for software testing where the whole process, including the testing staff, is provided as a cloud service.

As mentioned earlier in this subsection, currently there are many cloud service models in the market. This is because different cloud providers use different service models to reflect the types of services they provide. However, the differences between cloud service models are minute. In addition, these models complement each other. The most dominant service models currently in use are the ones discussed in this subsection.

As a final note, the reader should not mix cloud service models with cloud reference models and frameworks. While the former are concerned with classifying services at higher levels of abstraction into layers or service classes, the latter are more detailed frameworks that relate services to the different cloud models. The next subsection discusses cloud reference frameworks, models, and architectures in more detail.

TABLE II
5W + H QUESTIONS TO INVESTIGATE CLOUD COMPUTING REFERENCE FRAMEWORKS.

What	What is a Cloud Computing reference model/architecture/framework?
Who	Who needs a Cloud Computing reference framework?
Why	Why is a Cloud Computing reference framework required?
When	When will a Cloud Computing reference framework be available?
Where	Where are Cloud Computing reference frameworks being developed?
How	How do Cloud Computing reference frameworks differ from each other?

3. Cloud Computing Reference Models, Architectures, and Frameworks

Standardization is currently a vibrant concern in Cloud Computing. Creating reference models, architectures, and frameworks for Cloud Computing was of top priority in the Cloud Computing agenda of 2009–2011. For this reason, this section will investigate the Cloud Computing reference models, architectures, and frameworks by applying the six honest serving men,[2] or 5W + H questions to get a clear understanding of *what* are reference models (RMs), reference architectures (RAs), and reference frameworks (RFs), *who* needs them and *why*, *when* these frameworks and models will be available, *where* these reference models are being developed, and finally, *how* these frameworks differ from each other? Table II summarizes these questions.

3.1 (What) The Definition of Cloud Computing Reference Model, Architecture, and Framework

In software engineering, a reference model (RM) is an abstract, conceptual, technology independent framework, which represents a set of domain concepts and the relationships between them. It is usually used by domain experts who are working independently toward a standard [23]. When reference model concepts are arranged in a specific order (pattern) to provide a specific solution for recurrent problem, the generated architecture is called a reference architecture (RA) [24]. Together, the set of RMs and RAs create a reference framework (RF). While it is important to use the correct terminology when describing models, architectures, and frameworks,

[2] Six questions What, Where, Who, When, Why, and How, called 5W1H, from “Six Honest Men” poem of R. Kipling, Just so stories. Penguin Books, London, 1902.

formality is absent from several works that have been recently published in the domain of Cloud Computing [25]. For example, both IBM and NIST Cloud Computing Reference Architectures are more reference frameworks than reference architectures.

Cloud Computing reference models (CCRM) abstract the Cloud Computing concepts, mainly in terms of services, service models, capabilities, and roles; and define relationships between these concepts. A CCRM can be generic or specific to a particular sub-domain (e.g., development, security) within the Cloud Computing domain. Creating a cloud reference model involves two main tasks: first, identifying service types (i.e., security, management, monitoring) and the layer they belong to; and second, determining the different types of relationships with the identified services such as, the relationships between the services themselves, the relationships between the identified services and the cloud stakeholders (actors), and the relationships between the services and other cloud models (i.e., service models, deployment models, etc.).

The service models discussed in the previous subsection make up only one dimension within a Cloud Computing Reference Framework (CC RF). A CC RF is more sophisticated, because it does not only focus on service layers and their relationships, but also on services within each layer and their interaction relationships with service consumers and providers. For example, the PaaS layer consists of several types of services (e.g., management and monitoring). These services can be categorized into subgroups based on different factors, such as common features or similar policies that can be applied to them (e.g., who is allowed to access a service, and when and how the access can be made). Services belonging to the same policy are usually grouped in the same category.

3.2 (Who) Cloud Computing Reference Framework Users

A Cloud Computing RM/RF/RA is vital for all cloud stakeholders. In fact, many Cloud Computing reference frameworks such as the NIST CCRA and the IBM CCRA devoted an integral part of the Framework to explain the roles of each stakeholder within the Cloud Computing domain. A list of potential beneficiaries from reference frameworks and models includes the following:

- *Standardization bodies:* Organizations and industry consortiums that create the cloud standards.
- *Cloud creator:* Organizations or individuals who build cloud services; a cloud creator is not necessarily a cloud *provider* [21].
- *Cloud provider:* An individual or organization that makes one or more cloud services available to consumers based on a Service Level Agreement (SLA). In

several scenarios, a cloud provider can also be a consumer of services provided by another provider, or creator for their own services.

- *Cloud consumer:* An individual or organization that acquires, uses, and manages a cloud service that is provided by a cloud service provider directly or through a broker [26]. A cloud consumer can be a system administrator, a developer, or an end user (i.e., business process manager) depending on the service level (i.e., IaaS, PaaS, SaaS). Some cloud consumers are providers.
- *Cloud carriers:* A party that provides connectivity between cloud *providers* and *consumers*.
- *Cloud brokers:* An intermediary that negotiates the relationship between cloud providers and consumers [26].
- *Cloud regulators:* Legislation bodies and enforcement agencies that are responsible for enacting laws and creating regulations for Cloud Computing.
- *Cloud auditors:* A third party (usually governmental) that conducts an independent assessment of the cloud implementation to make sure that it is in adherence to regulations and standards [27].
- *Cloud educators:* Institutions and individuals who provide education materials and teach others about Cloud Computing.
- *Cloud learners:* Students, developers, or anyone who is interested in learning about Cloud Computing.

Each of the aforementioned beneficiaries makes use of the Cloud Computing reference framework in a different way. The next subsection explains why cloud stakeholders needs a cloud reference framework.

3.3 (Why) The Need for Cloud Computing Reference Frameworks

There are several reasons for creating cloud reference models and frameworks. As explained earlier, reference models are used by *domain experts* and *standardization bodies* to exchange domain knowledge in implementing a standard.

Reference frameworks are helpful for organizations while implementing or adopting a new technology. Cloud architecture reference models provide consistent and coherent architectural representations, and fundamental principles that can be used in the design and implementation of different projects. By having a cloud reference framework, *cloud providers* or their *creators* can apply best practises and guidelines to build their own clouds. Reference models can help *providers* focus on their distinctive advantages while making sure that they implemented all of the essential components. Furthermore, *cloud providers* can address interoperability issues with other providers at the early stages of the implementation, by providing support for the required interfaces for inter-cloud communications.

On the other hand, *cloud consumers* can use the cloud reference frameworks as part of their strategic planning and risk analysis. Reference frameworks can help consumers to understand the different *providers*' offers, in order to select the one that satisfies their needs. Similarly, *consumers* can measure risks such as the consequences of the lack of portability. Consumers should have access to reference frameworks and Service Level Agreements (SLAs), in order to realize their responsibilities and obligations, so as to understand what they should expect from a *cloud provider* and what they should implement themselves in order to fully harness Cloud Computing advantages like availability and scalability. This can help the *consumers* optimize resource utilization and reduce service operation costs.

Cloud reference models can also help *developers* effectively exploit services. For example, cloud specific reference models such as the one proposed in the paper "A Reference Model for Developing Cloud Applications" [28] can help cloud application developers understand cloud application requirements, and can guide them toward better architectural decisions that are independent of any cloud development environment. Such a reference model can boost the developers' learning curve and increase their productivity by providing them with the main principles of how to effectively reuse existing components and services in designs and implementations.

Cloud reference frameworks and models are used to educate *learners*, help *educators*, and improve communication between all cloud stakeholders. Reference frameworks clearly define the responsibilities and roles of all Cloud Computing actors.

3.4 (When and Where) The Status of Cloud Architecture Reference Models

Despite their roles, all cloud stakeholders (i.e., providers and consumers, corporate, and individuals) are interested in cloud reference models. Nevertheless, currently there is no complete or standardized cloud architecture reference model available. Cloud reference models are in the development stage, and are expected to stay in this stage as long as the Cloud Computing model is evolving. However, the Cloud Computing industry and academia are currently in the process of investigating standard reference models and frameworks for Cloud Computing.

In 2010, Forrester Research was able to count more than 75 different parties working on cloud standards [29]. Today, the number is expected to be much more than that. Different groups, standardization bodies, and industrial consortiums (e.g., Distributed Management Task Force (DMTF), Open Grid Forum (OGF), The Institute of Electrical and Electronics Engineers (IEEE), Open Group, Cloud

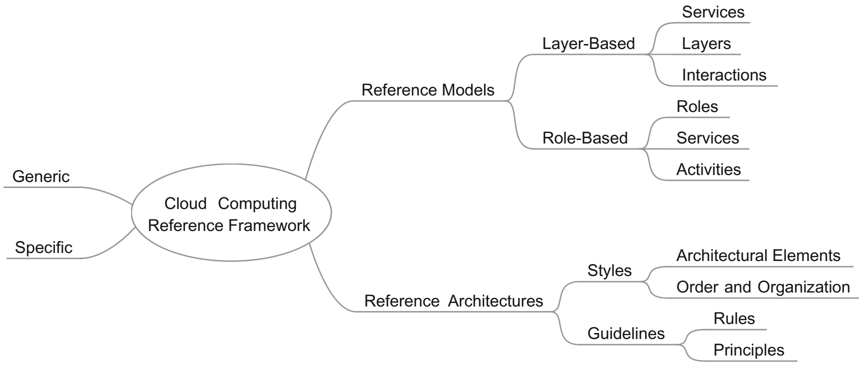


FIG. 3. Cloud Computing reference frameworks classifications.

Security Alliance (CSA), Open Cloud Consortium (OCC), OpenStack, and Object Management Group (OMG)), in addition to individual cloud providers (e.g., IBM, Microsoft, HP, Cisco, and CA), international agencies (e.g., ITU), federal agencies (e.g., GSA) and academic research groups [30,31,19,28,32], are all working in parallel to define standardized cloud reference models, frameworks, and architectures.[3] The next subsection will highlight some of the differences between these frameworks, models, and architectures.

3.5 (How) A high-Level Comparison Between Cloud Computing Reference Frameworks

As explained in the previous subsection, different Cloud Computing reference frameworks, models, and architectures are either developed or are currently being developed. The question is how these frameworks differ from each other, and how to select the framework or model that best fits your needs.

Figure 3 is a classification of Cloud Computing reference frameworks based on the current frameworks in the market. The most obvious distinction between current reference models and frameworks is that some of them are generic while others focus on specific areas. The NIST has recently compared six generic Cloud Computing Reference Architectures and five specific ones [33].

The analysis of current reference frameworks reveals that these frameworks can be decomposed first, into reference models, which consists of architectural elements and their relationships; and second, into architectural styles and principles, which

[3] This list is just exemplary and is not comprehensive.

represents a set of rules and guidelines that ensure enforcing best practices in methodical ways.

Most current frameworks focus on reference models. Almost all of these frameworks draw relations between services and either all or some of the following architectural elements:

- *Roles:* The main actors that communicate with the cloud service (i.e., Cloud Service Consumer, Provider, or Creator).
- *Activities:* Those actions and capabilities that connect a role to a service (i.e., integrate, consume, manage, provide).
- *Layers:* A classification of similar services based on finding commonalities. For example, based on the technology stack a service can be classified into infrastructure, middleware, application, and business process service.

The previous discussion leads us to another area of distinction between current cloud reference models and frameworks. Cloud Computing reference models and architectures can be either Role-Based or Layer-Based [25]. According to Wilkes [25], Role-Based CCRM maps services and activities to roles. Examples of role-based frameworks include: the DMTF Cloud Service reference architecture [34], the IBM Cloud Computing CCRA [21] and the NIST CCRA [26]. Almost all role-based reference frameworks recognize the Cloud Service Provider and Cloud Service Consumer roles. Other roles depend on the framework's level of detail.

On the other hand, Layer-Based CCRM maps services and activities into layers. For example Cisco Cloud Reference Architecture Framework [35] consists of the following five layers, the datacenter technology architecture layer, security layer, service orchestration layer, service delivery and management layer, and finally, the cloud services consumers layer. Similarly, the Cloud Security Alliance Cloud Reference Model (CSA-CRM) [27] shows the cloud stack as layers, and the IEFT Cloud Reference Framework (IEFT-CRF) [36] shows four horizontal layers and one stack vertical layer, and explains the capabilities within each layer in depth.

IBM released their second version of their CCRA on February 2011. Similarly, NIST proposed the first draft of their CCRA on March 2011 [33]. Figures 4 and 5 show the IBM Cloud Computing Reference Architecture and the NIST Cloud Computing Reference Architecture, respectively.

As shown in Fig. 4, the IBM CCRA consists of three main roles: the cloud service *consumer*, *provider*, and *creator*. Each of these roles consists of a different set of architectural elements, each of which consists of subcomponents and services. The most noticeable element in this framework is the Cloud Computing Management Platform (CCMP) [37]. The CCMP provides a set of business support

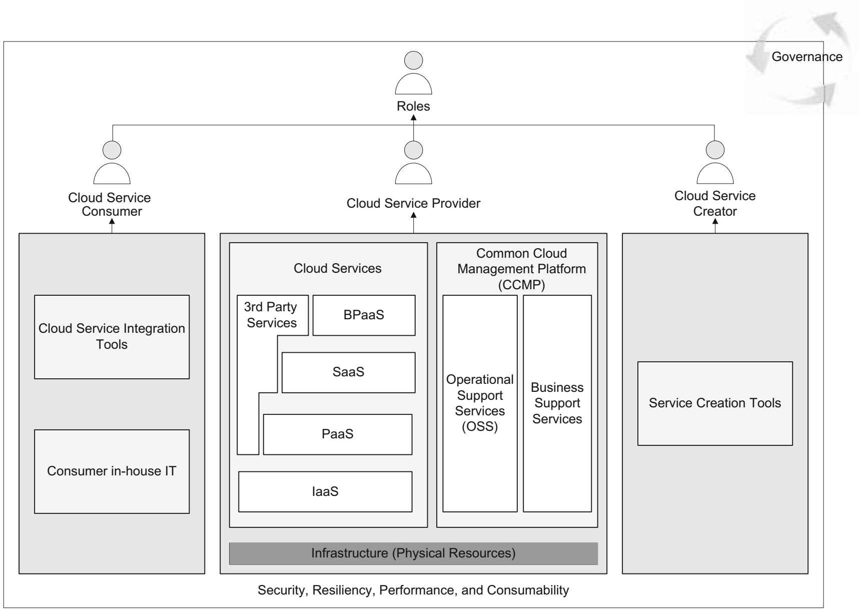

Governance
Roles
Cloud Service Consumer
Cloud Service Provider
Cloud Service Creator
Cloud Service Integration Tools
Consumer in-house IT
Cloud Services
3rd Party Services
BPaaS
SaaS
PaaS
IaaS
Common Cloud Management Platform (CCMP)
Operational Support Services (OSS)
Business Support Services
Infrastructure (Physical Resources)
Service Creation Tools
Security, Resiliency, Performance, and Consumability

FIG. 4. An overview of the IBM Cloud Computing Reference Architecture (adapted with modifications from [21]). Copyright IBM, Use with permission of IBM.

services (BSS) and operational support services (OSS) management services. CCMP services facilitate the communication between the different roles and the provider's underlying services (i.e., Infrastructure, Platform, Software and Business Process as a Service), through well-defined interfaces (APIs). For example, the OSS are managed operational services (i.e., provisioning, monitoring, and automation management services) that can be used by cloud service *consumers* to manage the cloud environment, by the cloud service *provider* to manage the infrastructure, and by the cloud service creators to create new services. Similarly, BSS are managed business services (i.e., billing, pricing, metering services), which are required by cloud service *creator*s to implement a cloud service. In addition to these roles and services, the IBM CCRA defines a set of four architectural principles and guidelines: namely, *Efficiency*, *Lightweightness*, *Economies-of-scale*, and *Generality*. The goal of these principles is to provide a set of best practices that can guide any cloud implementation.

Likewise, the NIST CCRA consists of five roles: the cloud service *consumer*, *provider*, *auditor*, *broker*, and *carrier*. Each role is associated with a set of activities; these activities connect the role to a set of components or services and their subcomponents. A closer look at NIST CCRA [26] shows that the NIST reference model has been inspired by the concepts extracted from their 2010/2011 survey of cloud architecture reference models [33]. For example, the Cloud Provider service orchestration was inspired by the Cisco Cloud Reference Architecture [35], which in turn, was inspired by the service-oriented architecture. Likewise, NIST cloud auditor role was inspired by the Open Security Architecture (OSA) model [38], which mainly focuses on cloud security.

There are lots of similarities between NIST and IBM CCRAs. While some architectural components are exactly the same, others have different wordings or are implicitly contained within other architectural components. Both NIST and IBM CCRAs are comprehensive frameworks. However, this does not mean that they are complete; in fact, there is no complete cloud reference framework to date. One of the strengths of the IBM CCRA is that it goes one step further and identifies some of the Cloud Computing architectural principles and guidelines as well as the Cloud Computing Management Platform (CCMP). Other reference models focus only on the relations among cloud services as well as the roles of the different cloud stakeholders. While identifying the roles, services, layers, and their relationships is important, a reference framework cannot be completed without the set of architectural styles that identify the different scenarios and best practices in a methodical way.

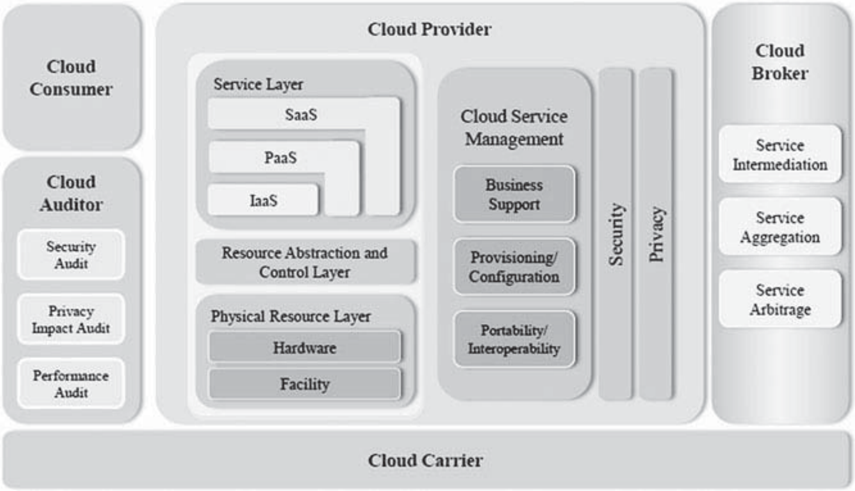


FIG. 5. NIST Cloud Computing Reference Architecture overview [26].

4. Cloud Computing's Relationship with Other Computing Paradigms

Cloud Computing is where the evolution lines of the service-oriented, grid computing, parallel computing, utility computing, autonomic computing, and virtualization paradigms meet. Cloud Computing is where the vision of all the previous technologies put into practice to meet the demands of high utilization and rapid change. The relationships between Cloud Computing and the aforementioned paradigms are strong to the extent that it is difficult in many cases to distinguish whether the cloud was really the big new thing, or if it is the same as any of these technologies. These relationships became more and more complex as the technologies reached maturity, and as the year of the cloud booming approached in 2008. The goal of this section is to explain how Cloud Computing has emerged from these technologies, and to clarify where these technologies belong within the context of Cloud Computing. This Section explains in detail the relationship between Cloud Computing and the service-oriented architecture, grid computing, parallel computing, utility computing, autonomic computing, and virtualization technologies.

4.1 Service-Oriented Architecture

The difference between Service-Oriented Architecture (SOA) and Cloud Computing is one of several repeatedly asked questions. This subsection tries to clear this ambiguity by explicitly explaining the relationship between the two paradigms.

The open group defined SOA as an architectural style that supports service orientation, where "Service orientation is a way of thinking in terms of services, and service-based development and the outcome of services" [39]. On the other hand, the NIST definition of Cloud Computing is based on the three service models that describe the classes of services provided, the four deployment models that provide different scenarios of service deployment, and the five distinctive characteristics that determine the basic requirements that a service must satisfy in order to be called a cloud service.

According to the previous definitions, Cloud Computing supports service orientation [21]. Service orientation provides foundations for Cloud Computing that enable global access, and ease of integrating different services and resources at run time, independent of the programming language used to implement the service. As in SOA, cloud services leverage network-based software through standards-based interfaces [40]. Today, it has become a norm to implement cloud services based on the Representational State Transfer (REST)-style architectures [41]. Cloud Computing services must also support scalability and guarantee Quality of Service (QoS). As we explained earlier in [28], SOA is an umbrella that describes any kind of service. A Cloud Computing service is a SOA service [42]; however, a SOA service is not necessarily a Cloud Computing service. A Cloud Computing service must satisfy all the Cloud Computing characteristics as defined in the NIST definition, which are optional in the case of generic SOA services. Cloud services follow specific standards, run under a specific environment, and are restricted by the cloud domain's technical and business boundaries. While SOA solutions need to implement all cloud characteristics from scratch if needed, Cloud Computing environments provide all of the tools required for creating and automatically managing and deploying services that adhere by default to the Cloud Computing characteristics. The service developer should not worry about service scalability, guarantee of service, on-demand service provisioning, or cost optimization.

SOA is mainly a business model that addresses business process management. However, cloud architecture addresses many technical details that are environment specific, making it more technical model. Cloud Computing is one realization of SOA. The relationship between Cloud Computing and SOA is similar to the relationship between Web-Services and SOA: Both are technologies that implement service orientation; however, Cloud Computing is more complicated than Web-Services. This is because a Web-Service is only one kind of cloud service that is usually provided by the software as a service layer.

Recognizing the relationship between SOA and Cloud Computing is essential for reusing the best practices and experience gained in the domain of SOA in the past years [21]. For example, instead of building a cloud ontology or modeling language from scratch, current SOA modeling languages (e.g., SOAML [43]) and ontologies can be used [44]. IBM, for instance, used the Open Group SOA Reference Architecture [45] in their definition of the IBM Cloud Computing Reference Architecture (CCRA). Such reuse can assist in concentrating on cloud concerns instead of addressing all cross-cutting concerns with SOA. Similarly, Cloud Computing can reuse the latest SOA technologies related to Web-Services and Web 2.0 (e.g., rich Internet applications, mashups, AJAX, RSS), so as to define, discover, and implement cloud services.

4.2 Grid Computing

Cloud Computing evolved from Grid Computing, so as to address the problematic issues and fulfill the desired requirements that were impossible to fulfill with Grid Computing, due to its use of real physical hardware, operating systems, and applications distributed across the globe [46].

Cloud and Grid Computing are types of distributed computing, in which workloads can be distributed through a load balancer and assigned, in parallel if needed, to unused resources at run time. While workloads usually need to be known ahead of time in the case of Grid Computing, Cloud Computing introduced the concept of on-demand, in which resources can dynamically scale-out and -in based on the current workload. The on-demand feature provided organizations with the required agility to handle sudden and irregular increases or decreases in business demand. Thanks to the virtualization technology, the Grid Computing dream of a fixable pool of virtualized resources provided as a service on demand became a reality [47]. It is the power of virtualization and service orientation principles that made the big shift from the application oriented Grid Computing technology to the service oriented Cloud Computing technology possible. Note that the initiatives that used virtualization with Grid Computing preceded the concept of "on-demand" Cloud Computing. Examples of such initiatives includes the "Virtual Workspaces" in Grid [48] and Clusters [49]. The goal of the previous initiatives was to use virtualization to help achieve reliability and Quality of Service (QoS).

Reliability and QoS are the two main issues that Cloud Computing came to address in Grid Computing. Grid Computing resources are tightly coupled; a failure in one node may result in the failure of a series of nodes that depends on it. In Cloud Computing, resources are loosely coupled, which allows dynamic failover, or restarting nodes and applying different configurations for different applications at run time. This can help in creating portable as well as available applications. The cloud infrastructure considered availability from the first day; hence, it provided the basic requirements to design for failure, such as providing different failover levels (i.e., regions, zones, and nodes), monitoring capabilities, and measured resources.

On the other hand, it is difficult to guarantee quality of service (QoS) in Grid Computing [50]. This is because traditional Grid Computing does not provide centralized management for job scheduling and performance measurement. An individual user or virtual organization (VO) activity can impact the performance of other users using the same platform [47]; this will result in variable throughput and response time. Conversely, Cloud Computing guarantees the bandwidth and response time of services through centralized management and measured services; this makes Cloud Computing more suitable for mission critical systems.

In their paper, Buyya et al. [51] gave a comprehensive comparison of Cloud and Grid Computing based on almost 20 characteristics. The summary of this comparison is as follows: Cloud Computing is more fixable in terms of scalability and cheaper than Grid Computing, because it uses commodity computers instead of high-end servers. Unlike Grid, Cloud utilizes utility computing and hence supports pricing strategies based on supply and demand. Cloud uses virtual resources such as Virtual Machines (VMs) and hypervisors, in addition to multiple lightweight operating systems, compared to the real resources and standard operating systems used by Grid. Cloud Computing resources and services are more secure and easier to access and manage. Cloud applications use Web-services standards (i.e., SOA and REST), while Grid uses the Open Grid forum standards for communication; this makes cloud services more interoperable with other Web applications and services. Moreover, cloud applications are easier to integrate; accordingly, they have a higher potential for building third-party value added solutions. Although both Cloud and Grid support Service Level Agreement (SLA) negotiation, Cloud Computing can guarantee QoS and support failure recovery better than Grid. Finally, Cloud Computing supports a wider variety of applications from content delivery to scientific applications, whereas Grid is more suitable for high-throughput scientific applications.

4.3 Parallel and High-Performance Computing

Several cloud definitions described Cloud Computing as a type of parallel computing. For example, Buyya et al. [51] have given the following definition "Cloud is a parallel and distributed computing system consisting of a collection of interconnected and virtualized computers." The previous section discusses the relation between Cloud and Grid computing, and indicates that both are distributed systems. This subsection clarifies the relationship between parallel and Cloud Computing.

Parallel computing is a form of computation that exploits multiple computing resources at the same time to solve a large computational problem by dividing it into smaller computational problems that can be solved concurrently [52].

Parallelism can be accomplished at different granularities and distribution levels. For example, parallelism can be at the bit level, at the instruction level, at the data level, or at the task level. On the other hand, the distribution of computational units on resources can be within a single machine, as is the case with multi-core and multi-processor, machines clusters, or supercomputers; or between multiple machines distributed around the globe, as in grids. In Cloud Computing, parallelism is exploited at the task and data levels, by distributed virtual machines that are not bound to a physical place.

In conventional parallel computing approaches, a parallel application is divided into tasks. These tasks are then distributed to be executed on compute clusters, supercomputers, or Grid infrastructures [53]. Raicu et al. [54] classified parallel applications based on the number of tasks distributed and the input size:

- Heroic MPI Tasks are characterized by a low number of tasks and a small input size (e.g., tightly coupled MPI applications).
- Data Analysis Mining are characterized by a low number of tasks but a large input data size (e.g., Map-Reduce [55]).

- Many Loosely Coupled Tasks are characterized by a large number of tasks with modest data size (e.g., Swift [56] and Falkon [57]).
- Big Data and Many Tasks are characterized by a large number tasks and a large input data size (e.g., Dryad [58] and Sawzall [59]).

To ensure the performance of parallel applications, tasks and their input data should be located in the same place (i.e., allocated to the same resource). This is usually referred to as the data locality principle [8]. In traditional parallel computing approaches, a task and its input data are allocated to the available resource (the same or a different resource, depending on the approach used) every time the task is executed [60]. This requirement may add overhead to the system, especially in virtualized and widely distributed environments such as Grid Computing. On the other hand, the lack of mechanisms for detecting node failure in traditional widely distributed systems, and in the technologies used for task distribution, have made resource availability a crucial concern.

Cloud Computing has contributed to solving the data locality and availability problems in compute/data intensive parallel applications, particularly, when the input data size accessed by the application increases. In such applications, efficiency requires both the data and the application to be within the same node or affinity group. Cloud Computing provides technologies and frameworks (e.g., Map-Reduce and Hadoop), which allow moving computation tasks to where the data is located, so as to perform data processing. Cloud platforms also support centralized control by periodically checking (heartbeat) available resources, and worker nodes, and reassigning failed jobs to other active healthy nodes [46].

Traditional parallel applications were run on high-performance computers (e.g., supercomputers and clusters), which are very expensive. Cloud Computing made such infrastructures available for ordinary people with affordable prices, by employing the concepts of resource sharing and virtualization. The side effect of this cost reduction is extra latency. This additional latency is due to the time needed to configure and customize the Virtual Machines (VMs) to be used to run particular tasks.

Cloud Computing allows users to provision resources fairly, easily, and quickly. Cloud Computing has improved the time needed for resource provisioning from days, in the case of traditional queue-based job scheduling systems, to minutes [60]. Furthermore, many cloud resources are currently available to the cloud community as open source. This includes hypervisors (e.g., Xen), parallel computing frameworks (e.g., Hadoop), and even platforms (e.g., OpenStack), which allow users to build and customize their own private clouds easily and efficiently.

While the Cloud programming models (e.g., Map-Reduce) that have emerged to utilize Cloud infrastructures create new trends in performing data/compute intensive parallel computing applications, there are limitations for using these models. Cloud Computing programming models are suitable for loosely coupled, transaction oriented, and latency insensitive (requires low latency) parallel applications that have a large number of tasks or large volumes of data. For example, Cloud Computing is not suitable for systems that require complex communication patterns [60].

Cloud Computing has provided tools and technologies for building parallel applications. Many case studies on different Cloud platforms have shown the applicability of cloud platforms to building data/compute intensive and scientific applications [60,61]. However, whether cloud technologies are the best solution for such applications is still an open question. Further research is required to address the questions of when and where Cloud Computing can contribute to solving parallel and distributed problems; what are the disadvantages of using cloud technologies; and how to overcome current cloud limitations, such as building parallel applications that require complex communication patterns.

4.4 Utility Computing

As explained earlier, Cloud Computing has borrowed lots of concepts from different domains. One important concept with a tight relationship with Cloud Computing is Utility Computing. There is a lot of confusion about the relationship between Utility and Cloud Computing. The concept of utility is confused with public utility, and utility computing with the pay-per-use and on-demand capabilities. While all of the previous concepts have strong relationships with each other, they are different.[4] Confusing these concepts can result in misunderstanding the definition of Cloud Computing. Therefore, this section provides a clear definition of Utility Computing and places it within the context of Cloud Computing.

According to the Merriam-Webster dictionary, a utility is “something useful or designed for use,” and is a “quality of state” that describes usefulness (e.g., if copying a file is useful, then it is a utility) [62]. Consequently, computer resources and their outcomes, which are offered or designed by a provider to be reused or to benefit a consumer, can be seen as utilities, or more precisely, as computing utilities. When a utility is provided as a service for the public it is called a public utility [62]. This is the same concept that McCarthy referred to in 1961 as *public* Utility Computing “If computers of the kind I have advocated become the computers of the future, then computing may someday be organized as a public utility just as the telephone system

[4] Note that electricity is not the only public utility, and capitalism is not the only economic system.

is a public utility. The computer utility could become the basis of a new and important industry."

Based on the previous discussion, Utility Computing can be defined as a computer-related General Purpose Technology (GPT) [63] that is provided as a service (e.g., the Internet). A utility computing service is a reusable service that aims to maximize the benefit to the consumers of the utility service, and to accommodate their needs while maximizing the providers' resource utilization. A utility computing service has nine characteristics as shown in Fig. 6; seven essential and two selective characteristics, based on the application domain and service model used.

The following essential characteristics are inspired by the Rappa utility service requirements [64]:

(a) *Requisite:* The benefit that a utility service provides to the consumers' must be a necessity; that is, without it consumers cannot fulfill their needs and achieve their goals (e.g., as water is a requisite for life, a storage service is a requisite for any software).
(b) *Generic:* A utility service is a generic service that all consumers with common needs can use (e.g., all software systems use storage).
(c) *Repeatable usefulness:* The usefulness of a utility service from the point of view of consumers with common needs should be the same.
(d) *Shared:* A utility service is shared between at least two parties: the provider and the consumer.
(e) *Easy to use:* A utility service must be easy to use by a consumer. For example, it should be easy to access and easy to manage (e.g., it should be plug and play).
(f) *Scalable:* A utility service must have the ability to accommodate consumer demand or adapt to these demands as they increase or decrease.
(g) *Measurable (metered):* A consumer should be able to determine the level of benefits gained from a utility service. A utility is a quality attribute. The measurement of which requires some metrics. These metrics are called utility values. A utility value is calculated using a utility function and is usually defined from the beneficiary's point of view. Examples include the number of jobs processed per second, waiting time, or the total impact of the service on the system.

Our classification for utility service characteristics distinguishes between the essential and selective characteristics. Selective characteristics can take different scenarios depending on the service model used and the service application domain. For example, while many describe utility services as pay-per-use services, this is not an essential characteristic; many utility services are provided to consumers for free (e.g., email services) based on subscription, or by other methods. The pay-per-use

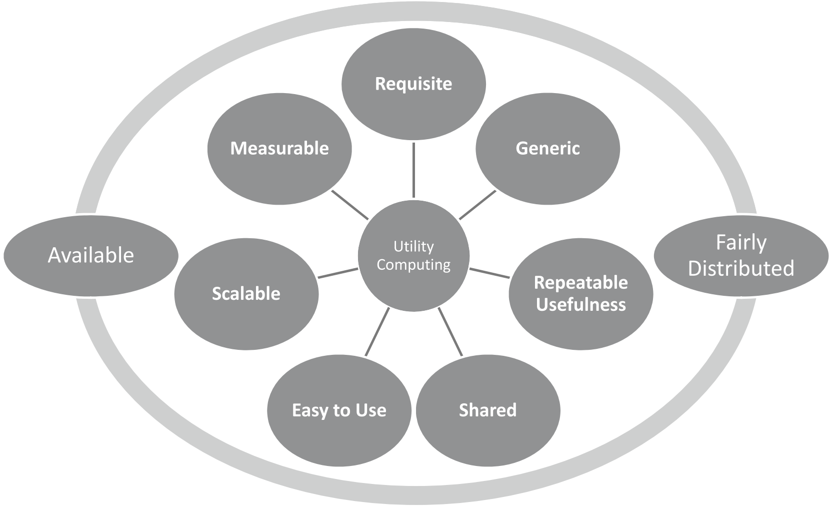


FIG. 6. Utility computing essential and selective characteristics.

model was considered as part of utility computing, as a result of thinking of utility computing in the context of autonomic computing and on-demand capabilities in 2002/2003 [65]. In fact, when researchers started actively using and employing the concept of utility computing in early 2000, the concept was completely decoupled from the pay-per-use model. The focus was instead on how to allow end users to share benefits, while leaving the payment method as an option. As pointed out by Karp et al. "people could pay for their computing by usage, modulated by guaranteed response requirements" [66]. Nevertheless, the utility service benefits still need to be metered to assist in ensuring fairness of utility service distribution. Similarly, while all utility services should be available, there are different mechanisms that consumers can use to request services. Finally, although all utility services must be generic, they are not necessarily public; a utility service can be deployed in any deployment environment. The following is the list of selective characteristics:

(a) *Fairly distributed*: A utility service must be fairly distributed to consumers. The distribution of a service depends on the business model used. While consumers will always ask for more resources and benefits, it is the decision of the provider to grant the service to the consumer that maximizes the yield. Without having accurate metrics to measure a benefit, the dominant business model will be either a borrow model, in which consumers who ask for the utility service first will receive it; or a flat rate fair share model, where all consumers receive the same benefits. While the borrow model may result in the starvation of some consumers, or the underutilization of resources due to greedy consumers, the flat rate fair share model does not take consumer needs into consideration (some consumers may need more resources). Accordingly, a utility service must enforce a fair distribution of utility to achieve maximum utilization. A fair distribution of utility services can be achieved by several scenarios such as the following:

(i) *Pay-per-use:* The pay-per-use scenario depends on supply and demand, and market share, so as to ensure that resources are distributed fairly. This scenario is based on the assumption that a consumer who is in more need for a service is more willing to pay for it. Similarly, a consumer who does not need a resource will release it to avoid unnecessary payments. This is the best solution for resource utilization from a provider's view point. Providers can guarantee that they have the most yields out of their resources based on the current market. However, a pay-per-use model may not be the best solution from a consumer's point of view. This is because consumer needs are not always combined with their ability to pay. Also, a pay-per-use model may not be the best solution in private and community deployment models, where the consumers (users) and providers share the same goals.

(ii) *Based on outcome impact:* The outcome impact of a utility service is one of the important factors to be considered when granting a utility service to a consumer. The generic rule is to reduce losses and increase benefits. A utility service should be granted to the consumer who needs it the most, in order to avoid losses, or to the consumer who can generate a higher yield out of the service. Focusing on the outcome impact is more difficult to achieve, but more fair from a consumer's point of view. On the other hand, focusing on outcome impact is not always the best solution, at least from a public provider's perspective. This model can also be applied in situations where bedding information is not available, such as in private or community scopes.

(b) *Available:* From a consumer's point of view, the utility service should be always available; however, how the service is made available depends on different scenarios. The following are the main two scenarios:

(i) *On-demand:* This is the most popular scenario in dynamic and elastic environments, making it ideal for Cloud Computing. The on-demand availability is the result of combining autonomic and utility computing.

(ii) *Pre-assigned quota:* The first known utility services were based on pre-assigned quota, in which a provider reserves resources for the consumer, based on a subscription contract.

As explained earlier in the Grid computing subsection, Cloud Computing evolved from Grid by providing solutions in order to fix the Grid problems. The main problem with grid is its lack of efficient mechanisms to guarantee reliability and QoS. The previous subsection introduced some solutions to these problems, and highlighted the fact that some of the Cloud fixes are by providing metered services and algorithms to address unexpected spikes in demand, so as to assure QoS. In fact, one of the Cloud Computing strategies for tackling reliability and QoS problems is through adopting the utility computing paradigm. Utility computing provides mechanisms for fair share resource distribution. It offers utility functions for generating normalized utility values for different utility services to measure the benefit gained by a consumer. It then applies mechanisms such as pay-per-use or priority, based on the urgency and impact of the service, to enforce fair share between consumers, avoid resource starvations, and maximize resource utilization [47].

4.5 Autonomic Computing

Computing systems, applications, and services are getting more complex, heterogeneous, and dynamic each day. Managing these systems is a daunting task and an error-prone process. Autonomic computing has been inspired by the human autonomic nervous system, and is used to manage such complex and sophisticated systems. The main goal of autonomic computing is to realize computer and software systems that can manage themselves with little or no human interaction. An autonomic system is a system that can adapt to changes, which can be triggered by a change in the system's own state (e.g., failure) or the state of its environment (e.g., external events) [67]. Accordingly, an autonomic system requires sensing mechanisms to sense changes (sensors), reaction mechanisms to respond to changes when they happen (effectors), and a decision making engine (autonomic manager) to make the correct assessment [67]. An autonomic system satisfies one or all autonomic self-* (read as self-star) properties, where self-* can be one of the following: self-configuring, self-healing, self-optimizing, and self-protecting.

The relationship between autonomic computing and Cloud Computing is based on mutualism. Cloud Computing infrastructures and platforms have been designed and built based on autonomic computing concepts, in order to reduce the complexity of managing the large and distributed cloud datacenters, increase resource availability, enhance flexibility, and ensure optimal utilization. Today, Cloud infrastructure

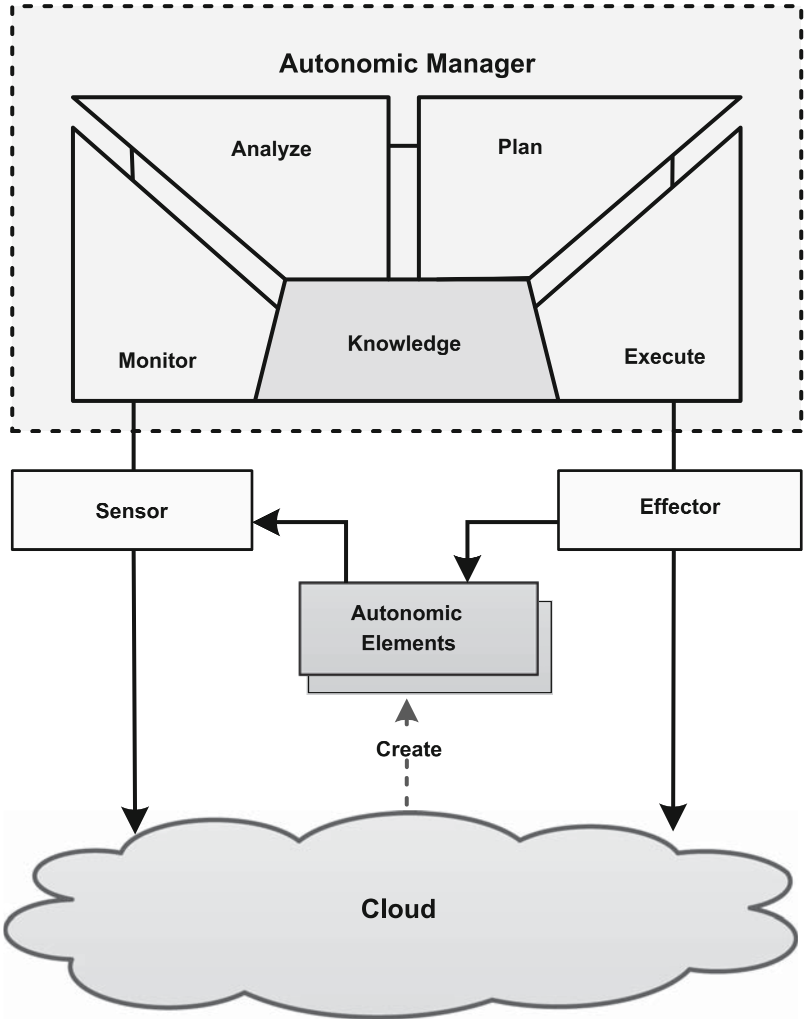


FIG. 7. The mutual relationship between cloud and autonomic computing.

services allow the automatic creation and migration of virtual machines, datacenter capacity control, proactive disaster recovery, and the dynamic provisioning of resources on demand and at run time, with minimum human supervision [47]. Cloud Computing platforms allow automatic job scheduling, migration, and synchronization, in addition to data management services that allow storage creation, management, and replication on the fly. Furthermore, the cloud has adopted concepts from autonomic computing to allow dynamic naming and resource localization, and to build on-going monitoring as an inherent cloud property.

On the other hand, according to van Renesse and Birman "Autonomic systems cannot be built simply by composing autonomic components. Autonomic system-wide monitoring and control infrastructures are needed" [68]. The Cloud Computing infrastructure provides everything needed for building autonomic systems and applications. Cloud Computing platforms provide services for sensing, analyzing, planning, and executing applications, and building knowledge about the system [69, 70]. Using cloud infrastructure resources and services, it is much easier to build fault-tolerant applications and to track them. Cloud infrastructures facilitate replicating applications to handle more user requests, reconfiguring them to work on new operating systems, and maintaining their operation near optimal performance. In fact, Cloud Computing resources are easier to monitor because all services are metered, it is easier to change the behavior of services at run time by reconfiguring them, because cloud service implementations are decoupled from their configuration, it is easier to replicate the services and hence tolerate failures, and finally the cloud provided services are easier to compose because of its virtual distributed service-oriented nature. This makes Cloud Computing an autonomic environment that is optimal for building autonomic systems.

Figure 7 shows the mutual relationship between cloud and autonomic computing. The figure shows that while cloud environments use autonomic managers to manage cloud resources, these environments are used to create autonomic systems and self-managed elements.

4.6 Virtualization

Virtualization is a main enabling technology for Cloud Computing. Virtualization is creating a temporarily simulated or extended version of computing resources (software or hardware) such as processors, operating systems, storages, and network resources. The simulated or extended version (virtual machine) will look like a real resource. Virtualization creates an abstraction layer between computing resources and the application that use them.

The goals of virtualization are first, to fully utilize the shared resources by applying partitioning and time-sharing techniques; second, to centralize resource management; third, to enhance datacenter agility and provide the required scalability and elasticity for on-demand capabilities; fourth, to improve testing and running software diagnostics on different operating platforms; fifth, to improve the portability of applications and workload migration capabilities; sixth, to provide the isolation required for a high degree of reliability, availability, security, and privacy; seventh, to enable server consolidation; and eighth, to provide a foundation for self-management frameworks.

In a nutshell, a virtualized ecosystem consists of virtual machines (VM), the virtualization technology used to create the virtual machines, and the virtual appliances or virtual images that run on top of the virtual machines. This subsection highlights the main virtualization architectures, technologies, and virtual appliance formats.

The idea of virtualization was established decades ago [71]. IBM created the first Virtual Machine Monitor (VMM) in 1965. The first IBM VMM allowed the host (IBM 360/65) to share its memory. In 1967, IBM created the first full virtualization VMM [72]. Since then, virtualization technologies have rapidly evolved. Several virtualization technologies currently exist. These technologies have different architectures and use diverse techniques for virtualization.

Figure 8 shows a classification of server virtualization technologies. In this classification, we focus on full virtualization, where the entire operating system can run on the virtual machine. It is important to notice that the term “full virtualization” also appears in the context of virtualization techniques, but with different semantics. Full virtualization, as opposed to paravirtualization, means the virtualization is achieved with no assistance of hardware or OS.

Server virtualization technologies play a key role in Cloud Computing. Cloud Computing has utilized virtualization in order to manage its large distributed datacenters and overcome most of the operational challenges of these datacenters.

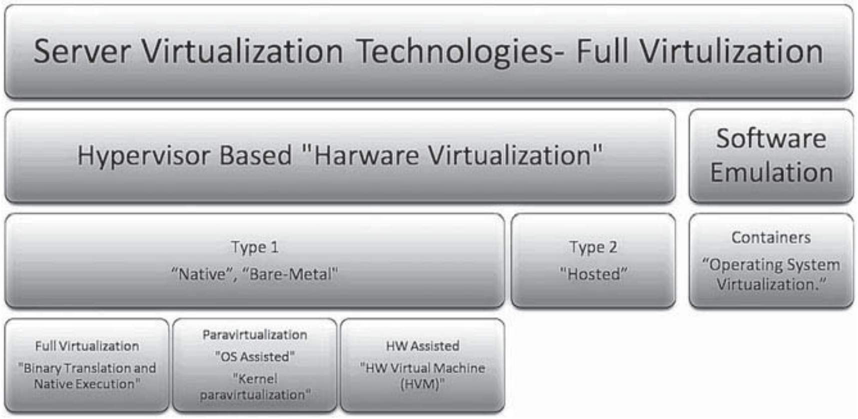

Server Virtualization Technologies- Full Virtulization
Hypervisor Based "Harware Virtualization"
Software Emulation
Type 1
“Native”, “Bare-Metal”
Type 2
"Hosted"
Containers
“Operating System Virtualization.”
Full Virtualization
"Binary Translation and Native Execution"
Paravirtualization
"OS Assisted"
"Kernel paravirtualization"
HW Assisted
"HW Virtual Machine (HVM)"

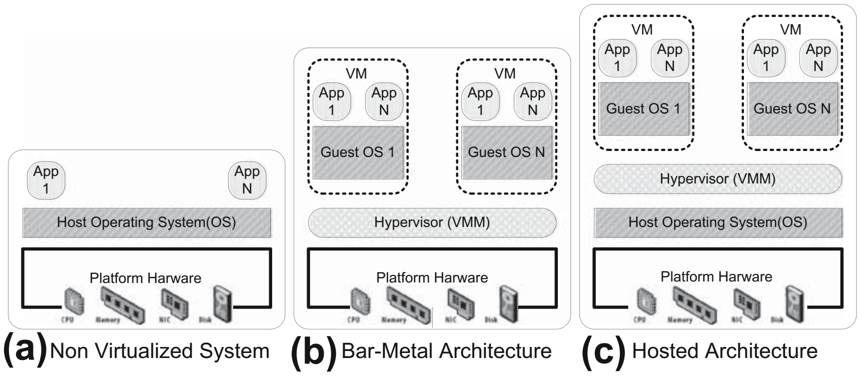
FIG. 8. Classification of sever virtualization technologies.

Resources in Cloud datacenters are shared between large numbers of users who have diverse needs and run different applications. Virtualization allows customizing, mounting, and allocating these resources to the users based on their needs. Virtualization enabled Cloud Computing to create a general virtual infrastructure and to provide the infrastructure as a service.

4.6.1 Virtualization Technologies

Depending on the location of the virtualization layer (hypervisor), there are two main hardware virtualization architectures: the bare-metal (Type 1) and the hosted (Type 2) architecture [73]. In the bare-metal architecture, the hypervisor is installed directly on the hardware. This architecture outperforms the hosted architecture, by allowing I/O devices to be partitioned into virtual machines for direct access. Bare-metal also has the advantage of supporting real-time and general purpose operating systems in parallel. However, since the hypervisor is installed directly on top of the hardware, it should include all device drivers. Furthermore, the lack of a base operating system makes the installation of these hypervisors more difficult and requires more customization and configuration.

On the other hand, the hosted architecture requires a base operating system to be installed first. The hypervisor (VMM) is installed on top of the hosting operating system. Hence, the VMM is easy to install and configure on most computers without the need for customization. However, a hosted architecture may result in performance degradation, because the I/O requests of the virtual machines need to be directed through the hosted OS. Another drawback of hosted architectures is their inability to



(a) Non Virtualized System (b) Bar-Metal Architecture (c) Hosted Architecture

FIG. 9. The architectures of a non-virtualized system, and both the bare-metal and the hosted virtualization systems.

run real-time operating systems directly inside the virtual machines. Figure 9 shows the simplified architectures of a non-virtualized system, and both the bare-metal and the hosted virtualization systems. The following are the main components of these systems:

- *Platform hardware:* The hardware resources, which are required to be shared.
- *Virtual machine monitor (VMM):* The program that is used to manage processor scheduling and physical memory allocation. It creates virtual machines by partitioning the actual resources, and interfaces the underlying hardware (virtual operating platform) to all operating systems (both host and guest).
- *Guest operating systems:* Guest operating systems are always installed on top of the hypervisor in hardware virtualization systems. Guest operating systems are isolated from each other. Guest OS kernels use the interfaces provided by the hypervisor to access their privileged resources.
- *Host operating system (optional):* The host operating system is the base operating system, under which the hypervisor is installed, in the hosted architecture case.

It is apparent from the previous discussion that both bare-metal and hosted architectures do not fully satisfy the Cloud Computing requirements. The bare-metal architecture is more popular in the Cloud Computing infrastructure because it is more efficient and delivers greater scalability, robustness, and performance. However, Cloud Computing still needs the hosted architecture's flexibility and compatibility.

4.6.2 Virtualization Techniques

There are several techniques for fully virtualizing hardware resources and satisfying the virtualization requirements (i.e., Equivalence, Resource control, and Efficiency) as originally presented by Popek and Goldberg [74]. These techniques have been created to enhance performance, and to deal with the flexibility problem in Type 1 architecture.

Popek and Goldberg classified the instructions to be executed in a virtual machine into three groups: privileged, control sensitive, and behavior sensitive instructions. While not all control sensitive instructions are necessarily privileged (e.g., x86). Goldberg's Theorem 1 mandates that all control sensitive instructions must be treated as privileged (i.e., trapped) in order to have effective VMMs.

Depending on the virtualization technique used, hypervisors can be designed to be either tightly or loosely coupled with the guest operating system. The performance of tightly coupled hypervisors (i.e., OS assisted hypervisors) is higher than loosely coupled hypervisors (i.e., hypervisors based on binary translation). On the other hand, tightly coupled hypervisors require the guest operating systems to be explicitly modified, which is not always possible. One of the Cloud infrastructure design challenges is to have hypervisors that are loosely coupled, but with adequate performance. Having hypervisors that are operating system agnostic increases system modularity, manageability, maintainability, and flexibility, and allows upgrading or changing the operating systems on the fly. The following are the main virtualization techniques that are currently in use:

(a) *Binary translation and native execution:* This technique uses a combination of binary translation for handling privileged and sensitive instructions[5] [74], and direct execution techniques for user-level instructions [74]. This technique is very efficient both in terms of performance and in terms of compatibility with the guest OS, which does not need to know that it is virtualized. However, building binary translation support for such a system is very difficult, and results in significant virtualization overhead [75].

(b) *OS assisted virtualization (paravirtualization)*: In this technique, the guest OS is modified to be virtualization-aware (allow it to communicate through hypercalls with the hypervisor, so as to handle privileged and sensitive instructions). Because modifying the guest OS to enable paravirtualization is easy, paravirtualization can significantly reduce the virtualization overhead. However, paravirtualization has poor compatibility; it does not support operating systems that cannot be modified (e.g., Windows). Moreover, the overhead introduced by the hypercalls can affect performance under heavy workloads. Besides the added overhead, the modification made to the guest OS, to make it compatible with the hypervisor, can affect system's maintainability.

(c) *Hardware-assisted virtualization:* As an alternative approach to binary translation and in an attempt to enhance performance and compatibility, hardware providers (e.g., Intel and AMD) started supporting virtualization at the hardware level. In hardware-assisted virtualization (e.g., Intel VT-x, AMD-V), privileged and sensitive calls are set to automatically trap to the hypervisor. This eliminates the need for binary translation or paravirtualization. Moreover, since the translation is done on the hardware level, it significantly improves performance.

Cloud providers have utilized different virtualization platforms to build their datacenters. The trend in the Cloud is to use platforms that combine paravirtualization and hardware-assisted virtualization to benefit from the advantages of both. The most notable VMM platforms are VMware, Xen, Hyper-V, and KVM. All these platforms

[5] *Note:* Sensitive instructions may not be privileged but still need to be trapped by the VMM.

use the Type 1 hypervisor based architecture. However, while VMware uses the direct driver model to install the hypervisor on bare-metal, the others use the indirect driver model. Moreover, all of these platforms support paravirtualization and full binary translation virtualization. Unlike in VMware, Xen, and Hyper-V the use of hardware-assisted virtualization, is mandatory in KVM. The reader can refer to [76] for a full comparison between these platforms.

4.6.3 Virtual Appliances

A virtual appliance is a Virtual Machine Image (VMI) that is designed and packaged to be deployed on a VM platform. A VMI is a self-contained package that consists of a software stack and metadata. The software stack consists of a set of pre-configured applications, the guest OS, middleware, libraries, containers, and all of the other required software, while the metadata represents information that can assist in the deployment and execution process of the VMI on the VM platform; for example, the support hardware needed to run the VMI (e.g., number of CPUs), the configuration variables (e.g., IP Address), and image constraints (e.g., 99.9% available).

Packaging applications in the form of virtual machine desk images can facilitate the distribution, installation, configuration, management, and execution of applications under an optimal VM platform. This also improves the application's portability and allows workload migration.

The variety of virtualization platforms used by various Cloud providers has hindered interoperability between virtual appliances, and has restricted the migration of applications and VMIs across Cloud platforms. This is because different virtualization platforms use different VM image formats. The lack of interoperability uncovered the need for a standard distribution format for virtual appliances. To tackle this issue, the Distributed Management Task Force (DMTF) created the Open Virtualization Format (OVF) in 2008.

The OVF is a vendor and platform independent standardized distribution format for virtual appliances. It defines the compatibility and optimization requirements of virtual appliances and allows VM distribution at a large scale. The OVF is portable and extensible in the sense that it allows adding features and customization by the DMTF or a third party. OVF sports the composition of multiple VMs, for example when a multi-tiered application workload is distributed into multiple VMs. In such cases, the VMI must include additional metadata related to the composition.

The OVF allows different products and vendors to exchange VMs. This allows the exchange of virtual appliances in the form of online marketplaces, which is one of the main goals of Cloud Computing.

4.7 Cloud Computing Relationships Summary

The sheer volume of knowledge that is required to merge technology streams has lead to separate but related technologies, which if accumulated, can result in the big new thing. This section shows an example of such a merger. After reading this section, it will become clear that Cloud Computing is the result of evolution and adoption of existing technologies and paradigms. The goal of Cloud Computing is to allow users to take benefit from all of these technologies, without the need for deep knowledge about or expertise with each one of them. The Cloud aims to cut costs, and help the users focus on their core business instead of being impeded by IT obstacles. Figure 10 summarizes how Cloud Computing is related to the discussed technologies.

The main enabling technologies for Cloud Computing are virtualization and autonomic computing. Virtualization abstracts the physical infrastructure, which is the most rigged component, and makes it available as a soft component that is easy

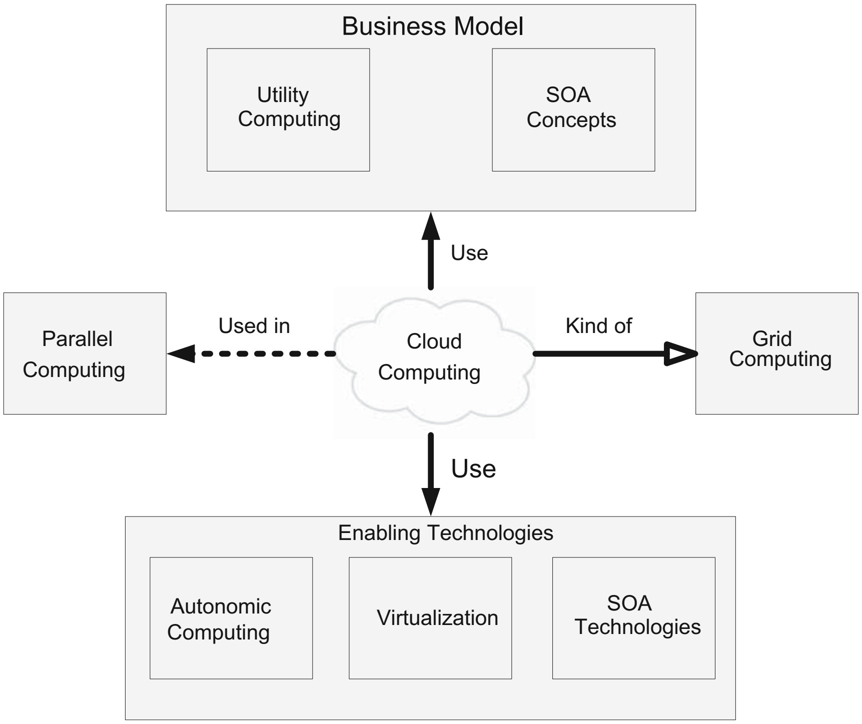


FIG. 10. Cloud Computing relationship with other computing paradigms.

to use and manage. By doing so, virtualization provides the agility required to speed up IT operations, and reduces cost by increasing infrastructure utilization. On the other hand, autonomic computing automates the process through which the user can provision resources on-demand. By minimizing user involvement, automation speeds up the process and reduces the possibility of human errors.

Users face difficult business problems every day. Cloud Computing adopts concepts from SOA that can help the user break these problems into services that can be integrated to provide a solution. Cloud Computing provides all of its resources as services, and makes use of the well-established standards and best practices gained in the domain of SOA to allow global and easy access to cloud services in a standardized way. Cloud Computing also utilizes concepts from utility computing in order to provide metrics for the used services, based on the benefits gained. These metrics are at the core of the pay-per-use model in public Clouds. Having measured services is also an essential part of completing the feedback loop in autonomic computing, which is required for fully automating services so that they can scale on-demand and automatically recover from failures.

Cloud Computing is a kind of Grid Computing; it has evolved from Grid by addressing the QoS and reliability problems. Cloud Computing provides the tools and technologies to build data/compute intensive parallel applications with much affordable prices compared to traditional parallel computing techniques.

5. Conclusions and Future Directions

This introductory chapter has discussed Cloud Computing's basic principles and reference frameworks, and has surveyed the main computing paradigms that gave birth to the Cloud Computing. The Cloud Computing paradigm is still evolving. New additions and refinements to the cloud models, characteristics, and technologies occur on a daily basis. Fortunately, the plethora of existing reference models, architectures, and frameworks, as well as those underdevelopment are a sign that Cloud Computing is on its way to maturity. These reference frameworks will motivate hesitated organizations to adopt Cloud Computing.

It is true that Cloud Computing is an evolution of existing technologies rather than a technological revolution. However, it is not merely a natural evolution of a single technology, but it is a hybrid-cross (intra-specific) between different technologies resulting in the next big thing, analogous to the liger, which is the hybrid-cross between a lion and a tigress. Cloud Computing fixes the problems with the technologies that it evolved from, and adds new desired characteristics by integrating technologies.

The benefits that organizations can obtain by adopting Cloud Computing are evident. Cloud Computing reduces both organizations' capital and operational expenses. Furthermore, Cloud Computing provides organizations with the agility required to cope with sudden and unexpected changes in business demand. For example, the high level of automation, flexibility, and scalability that the Cloud Computing infrastructures provide can reduce the time of deploying a new server from 7.5 weeks to 3 min as David M. Powers explained [77]. Cloud Computing inherently support disaster recovery and fault tolerance that organizations need to ensure business continuity. It also provides the high visibility required for collaboration across users and partners, by allowing resources to be accessed at anytime from anywhere when policies permit.

Cloud Computing is the future for years to come, if not for being a cutting-edge technology, then for the benefits it has yielded to the business and society through its promises of a smarter planet and a green environment (i.e., by reducing organization's greenhouse-gas emissions). Moreover, the fact that most organizations have invested or are currently investing in cloud technologies and solutions, is sufficient to prove that this technology is not dying soon, at least until organizations harvest their return of investment on Cloud Computing.

In the future, Cloud Computing will focus more on mobility. We are going to see more applications that connect to the cloud datacenters, from smartphones, vehicles, and TVs, to utilize the cloud capabilities. The cloud will shape our life and make it easier and smarter. For example, mobile devices will be able to perform tasks that are impossible without powerful computation power such as video encoding, voice processing, and image processing. Applications in smart devices will use the cloud to crunch huge data sets and exploit business analytics to provide faster and smarter decisions. For example, a car will be able to send information about its surroundings, the environment, as well as its current status. The information will be augmented with all the information sent from other devices to make smart decisions. This in our opinion will promote what we call the "Decision Support as a Service" industry. An industry that is based on collecting information, and then analyzing it to provide subscribers with accurate decisions as a service. We anticipate that the decision support as a service will have a significant impact on fields such as medicine, economy, and environmental science. We will be able to see smart medical diagnoses systems, automatic financial market brokers, and accurate climate-change forecasting.

The research and scientific community will use the cloud to solve complex problems; Cloud Computing will unleash the productivity and innovation in all scientific fields. However, before this stage, there will be a significant advancement in all the interleaving fields that affect Cloud Computing. For example, new parallel computation models (i.e., map reduce) that are more efficient and target different types of applications will be created, new standards, interfaces, and software stacks will be created to support applications' interoperability and data, workloads, and applications' portability. There will also be new frameworks that allow smart dynamic elasticity, and ensure service high availability at the enterprise level, which will depend on different cloud providers.

In the future, private clouds are expected to fade, while public and hybrid clouds will continue to be more popular. Public cloud will be the dominant model for small organizations and end users, while hybrid clouds will be the solution for medium-size and large (non-provider) organizations. The increase adoption of hybrid clouds, in addition to the increase use of cloud applications to empower mobile devices will be the impetus of a serious network traffic problem. This is due to first, the commuting workloads between the public cloud datacenters and both the private datacenters as well as the mobile devices, and second, because all communications usually happen through the Internet, which is slower, more expensive and less reliable than hard drives. Some providers currently allow data shipping on hard drives using couriers (e.g., FedEx) to solve this problem. However, this does not provide an efficient solution for the problem. In the future, there will be a need for protocols and mechanisms to address these difficulties. For example, someone may think of how common tasks can be factored as standalone components to reduce data traffic. Finally, in the future Internet providers will start pushing for laws to protect the scarce Internet bandwidth and the notion of the "Metered Internet" will become the norm.

Cloud Computing has been built on top of strong and mature technologies. However, it still lacks case studies and best practices on how to harness this technology, and to avoid the pitfalls that may result in immature design or inappropriate application. Cloud Computing has lots of areas to be investigated. We anticipate that this book chapter will provide researchers and practitioners an appropriate foundation to start their journey behind the cloud.

REFERENCES

[1] D. Parkhill, The Challenge of the Computer Utility, first ed. Addison-Wesley Pub. Co. 1966.

[2] L. Vaquero, L. Rodero-Merino, J. Caceres, M. Lindner, A break in the clouds: towards a cloud definition, ACM SIGCOMM – Computer Communication Review 39 (2008) 50–55.

[3] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R.H. Hatz, A. Konwinski, G. Lee, D.A. Patterson, A. Rabkin, M. Zaharia, Above the Clouds: a Berkeley view of cloud computing, Technical Report UCB/EECS-2009-2, Electrical Engineering and Computer Sciences University of California, Berkeley, 2009.

[4] J. Staten, T. Schadler, J.R. Rymer, C. Wang, Q&A: by 2011, CIOs must answer the question, why not run in the cloud? Technical Report, Forrester Research Inc., 2009. Retrieved from: <http://www.forrester.com/rb/Research/qn,/q/id/55193/t/2>.

[5] J. Staten, C. Kane, R. Whiteley, You're not ready for internal cloud, Technical Report, Forrester Research Inc., 2010. Retrieved from: <http://www.forrester.com/rb/Research>.

[6] P. Mell, T. Grance, The NIST definition of cloud computing, Recommendations of the National Institute of Standards and Technology Special Publication 800-145, National Institute of Standards and Technology, 2009.

[7] K. Tsakalozos, M. Roussopoulos, V. Floros, A. Delis, in: ICDCS'10, 30th IEEE International Conference on Distributed Computing Systems, IEEE, 2010, pp. 74–85.

[8] M. Zaharia, D. Borthakur, J. SenSarma, K. Elmeleegy, S. Shenker, I. Stoica, in: EuroSys'10, Fifth European Conference on Computer Systems, ACM, 2010, pp. 265–278.

[9] T. Haselmann, G. Vossen, S. Lipsky, T. Theurl, CLOSER'11, The First International Conference on Cloud Computing and Services Science, INSTICC (2011) 104–109.

[10] A. Marinos, G. Briscoe, Community cloud computing, Cloud Computing (2009) 472–484.

[11] G. Briscoe, A. Marinos, in: DEST'09, Third IEEE International Conference on Digital Ecosystems and Technologies, IEEE, 2009, pp. 103–108.

[12] G. Reese, The economics of cloud computing, Online, 2008. Retrieved from: <http://broadcast.oreilly.com/2008/10/the-economics-of-cloud-c.html>.

[13] B. Kepes, Moving your infrastructure to the cloud: how to maximize benefits and avoid pitfalls, White Paper, Diversity Limited and Rackspace, 2011. <http://www.rackspace.com/knowledge_center/whitepaper/moving-your-infrastructure-to-the-cloud-how-to-maximize-benefits-and-avoid-pitfalls>.

[14] J. Moore, Cloud computing economics: technical and business information about cloud computing, Online, 2009. Retrieved from: <http://www.cloudcomputingeconomics.com/2009/01/experience-curves-for-data-center.html>.

[15] A. Greenberg, J. Hamilton, D.A. Maltz, P. Patel, The cost of a cloud: research problems in data center networks, ACM SIGCOMM – Computer Communication Review 39 (2008) 68–73.

[16] M. Christodorescu, R. Sailer, D.L. Schales, D. Sgandurra, D. Zambonim, CCSW'09, First ACM Workshop on Cloud Computing Security, CCSW'09, ACM, New York, NY, USA (2009) 97–102.

[17] M. Dawson, Debunking the top three cloud security myths, Online, 2011. Retrieved from: <https://blog.cloudsecurityalliance.org/2011/03/30/debunking-the-top-three-cloud-security-myths/>.

[18] T. Micro, Top 5 myths of cloud computing, Online, 2010. Retrieved from: <http://cloudsecurity.trendmicro.com/top-5-myths-of-cloud-computing-security/>.

[19] W. Tsai, X. Sun, J. Balasooriya, in: ITNG'10, Seventh International Conference on Information Technology: New Generations, 2010, pp. 684–689.

[20] X. Li, L. Zhou, Y. Shi, Y. Guo, in: ICMLC'10, Second International Conference on Machine Learning and Cybernetics, vol. 6, IEEE, 2010, pp. 2843–2848.

[21] M. Behrendt, B. Glasner, P. Kopp, R. Dieckmann, G. Breiter, S. Pappe, H. Kreger, A. Arsanjani, Introduction and architecture overview IBM Cloud Computing Reference Architecture 2.0, 2011. Retrieved from: <https://https://www.opengroup.org/cloudcomputing/uploads/40/23840/CCRAIBMSubmission.02282011.doc>.

[22] L. Youseff, D.M.D. Silva, M. Butrico, J. Appavoo, Cloud Computing and Software Services: Theory and Techniques, CRC Press, 2011, pp. 1–16.

[23] A. EarlF. Long (Ed.), Software Engineering Environments, Lecture Notes in Computer Science, vol. 467, Springer, Berlin, Heidelberg, 1990, pp. 115–129.

[24] A.D. Giordano, Data Integration Blueprint and Modeling: Techniques for a Scalable and Sustainable Architecture, IBM Press, 2011, pp. 19–20.

[25] L. Wilkes, Cloud computing reference architectures, models and frameworks, Technical Report, Everware-CBDI Inc., 2011.

[26] NIST, Nist cloud computing reference architecture, version 1, 2011. Retrieved from: <http://collaborate.nist.gov/ twiki-cloud-computing/pub/CloudComputing/ReferenceArchitectureTaxonomy/NIST_CC_Reference_Architecture_v1_March_30_2011.pd>.

[27] C.S. Alliance, Cloud security alliance security guidance for critical areas of focus in cloud computing V2.16, Technical Report, Cloud Security Alliance, 2009.

[28] M. Hamdaqa, T. Livogiannis, L. Tahvildari, CLOSER'11, First International Conference on Cloud Computing and Services Science, INSTICC, 2011, pp. 98–103.

[29] C. Wang, S. Balaouras, L. Coit, Q&A: Demystifying cloud security. An empowered report: getting past cloud security fear mongering, Technical Report, Forrester Research, Inc., 2010.

[30] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, K. Nagin, I. Llorente, R. Montero, Y. Wolfsthal, E. Elmroth, J. Caceres, et al, The reservoir model and architecture for open federated cloud computing, IBM Journal of Research and Development 53 (2009) 4–10.

[31] L. Zhang, Q. Zhou, in: ICWS 2009, Seventh IEEE International Conference on Web Services, IEEE, 2009, pp. 607–616.

[32] S. Charlton, in: OOPSLA09, 14th Conference Companion on Object Oriented Programming Systems Languages and Applications, 2009, pp. 17–26.

[33] NIST, Cloud architecture reference models: a survey V2, Technical Report, National Institute of Standards and Technology, 2011.

[34] Architecture for managing clouds: desktop management task force cloud computing reference architecture, 2010. Retrieved from: <http://www.dmtf.org/sites/default/files/standards/documents/DSP-IS0102_1.0.0.pdf>.

[35] K. Bakshi, Cisco cloud computing – data center strategy, architecture, and solutions, Technical Report, Cisco Systems, Inc., 2009.

[36] B. Khasnabish, J. Chu, S. Ma, Y. Meng, P. Unbehagen, M. Morrow, M. Hasan, Cloud reference framework 01, 2011. Retrieved from: <http://tools.ietf.org/html/draft-khasnabish-cloud-reference-framework-00>.

[37] I.G. Services, Getting cloud computing right: the key to business success in a cloud adoption is a robust, proven architecture, White paper CIW03078-USEN-01, IBM Corporation, 2011. Retrieved from: <http://public.dhe.ibm.com/common/ssi/ecm/en/ciw03078usen/CIW03078USEN.PDF>.

[38] Sp-011: cloud computing pattern, 2011. Retrieved from: <http://www.opensecurityarchitecture.org/cms/library/patternlandscape/251-pattern-cloud-computing>.

[39] The Open Group, Service oriented architecture: what is SOA?, Online, 2011. Retrieved from: <http://www.opengroup.org/soa/source-book/soa/soa.htm>.

[40] Y. Wei, M.B. Blake, Service-oriented computing and cloud computing: challenges and opportunities, IEEE Internet Computing 14 (2010) 72–75.

[41] R. Fielding, Architectural styles and the design of network-based software architectures, Ph.D. thesis, University of California, Irvine, 2000.

[42] L. Wang, G. von Laszewski, A. Younge, X. He, M. Kunze, J. Tao, C. Fu, Cloud computing: a perspective study, New Generation Computing 28 (2010) 137–146, http://dx.doi.org/10.1007/s00354-008-0081-5.

[43] OMG, Service oriented architecture modeling language (SOAML) – specification for the UML profile and metamodel for services (UPMS), 2009. Retrieved from: <http://www.omg.org/SoaML/1.0/Beta2/PDF>.

[44] B. Elvester, A.-J. Berre, A. Sadovykh, CLOSER'11, First International Conference on Cloud Computing and Services Science, INSTICC (2011) 98–103.

[45] A. Arsanjani, N. Kumar, SOA reference architecture, 2009. Retrieved from: <http://www.opengroup.org/projects/soa-ref-arch/uploads/40/19713/soa-ra-public-050609.pdf>.

[46] I. Foster, Y. Zhao, I. Raicu, S. Lu, in: GCE'08, Fourth Grid Computing Environments Workshop, IEEE, 2008, pp. 1–10.

[47] W. Voorsluys, J. Broberg, R. Buyya, Cloud Computing: Principles and Paradigms, Wiley, 2011, pp. 3–41.

[48] K. Keahey, I. Foster, T. Freeman, X. Zhang, Virtual workspaces: achieving quality of service and quality of life in the grid, Scientific Programming 13 (2005) 265–275.

[49] J. Chase, D. Irwin, L. Grit, J. Moore, S. Sprenkle, in: HPDC'03, 12th IEEE International Symposium on High Performance Distributed Computing, IEEE, 2003, pp. 90–100.

[50] J. Myerson, Cloud computing versus grid computing-service types, similarities and differences, and things to consider, Online, 2008. Retrieved from: <http://www.ibm.com/developerworks/web/library/wa-cloudgrid>.

[51] R. Buyya, C. Yeo, S. Venugopal, J. Broberg, I. Brandic, Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility, Future Generation Computer Systems 25 (2009) 599–616.

[52] V. Kumar, A. Grama, A. Gupta, G. Karypis, Introduction to Parallel Computing: Design and Analysis of Algorithm, vol. 110, Benjamin-Cummings, 1994.

[53] I. Foster, C. Kesselman, S. Tuecke, The anatomy of the grid: enabling scalable virtual organizations, International Journal of High Performance Computing Applications 15 (2001) 200–222.

[54] I. Raicu, Z. Zhang, M. Wilde, I. Foster, P. Beckman, K. Iskra, B. Clifford, SC'08, 21st ACM/IEE Annual Supercomputing Conference, SC'08, IEEE Press, Piscataway, NJ, USA (2008) 22:1–22:12.

[55] J. Dean, S. Ghemawat, MapReduce: simplified data processing on large clusters, Communications of the ACM 51 (2008) 107–113.

[56] Y. Zhao, M. Hategan, B. Clifford, I. Foster, G. von Laszewski, V. Nefedova, I. Raicu, T. Stef-Praun, M. Wilde, Swift: fast, reliable, loosely coupled parallel computation, in: IEEE Congress on Services, 2007, pp. 199–206.

[57] I. Raicu, Y. Zhao, C. Dumitrescu, I. Foster, M. Wilde, SC'07, Second ACM/IEE Annual Supercomputing Conference, SC'07, ACM, New York, NY, USA (2007) 43:1–43:12.

[58] M. Isard, M. Budiu, Y. Yu, A. Birrell, D. Fetterly, Dryad: distributed data-parallel programs from sequential building blocks, SIGOPS Operating Systems Review 41 (2007) 59–72.

[59] R. Pike, S. Dorward, R. Griesemer, S. Quinlan, Interpreting the data: parallel analysis with Sawzall, Scientific Programming 13 (2005) 277–298.

[60] J. Ekanayake, G. Fox, Cloud Computing, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering vol. 34, Springer, Berlin, Heidelberg (2010) 20–38, http://dx.doi.org/10.1007/978-3-642-12636-9_2.

[61] W. Lu, J. Jackson, R. Barga, HPDC'10, 19th ACM International Symposium on High Performance Distributed Computing, HPDC'10, ACM, New York, NY, USA (2010) 413–420.

[62] Dictionary and Thesaurus – Merriam-Webster Online, The definition of utility, 2011. Retrieved from: <www.merriam-webster.com/dictionary/utility>.

[63] T. Bresnahan, M. Trajtenberg, General purpose technologies engines of growth?, Journal of Econometrics 65 (1995) 83–108.

[64] M. Rappa, The utility business model and the future of computing services, IBM Systems Journal 43 (2004) 32–42.

[65] J. Kephart, D. Chess, The vision of autonomic computing, Computer 36 (2003) 41–50.

[66] A. Karp, R. Gupta, G. Rozas, A. Banerji, The client utility architecture: the precursor to E-speak, Technical Report, Technical Report HPL-2001-136, Hewlett Packard Laboratories, 2001.

[67] A. Ganek, Autonomic Computing: Concepts, Infrastructure, and Applications, CRC Press, 2007, pp. 3–18.

[68] R. van Renesse, K.P. Birman, Autonomic computing – a system-wide perspective, Autonomic Computing: Concepts, Infrastructure, and Applications, 2006, pp. 1–11.

[69] M. Huebscher, J. McCann, A survey of autonomic computing degrees, models, and applications, ACM Computing Surveys 40 (2008) 1–28.

[70] P. Horn, Autonomic computing: IBM's perspective on the state of information technology, Computing Systems 15 (2001) 1–40.

[71] R. Creasy, The origin of the VM/370 time-sharing system, IBM Journal of Research and Development 25 (1981) 483–490.

[72] R. Rose, Survey of system virtualization techniques, Technical Report, 2004.

[73] K.L. Kroeker, The evolution of virtualization, Communications of the ACM 52 (2009) 18–20.

[74] G. Popek, R. Goldberg, Formal requirements for virtualizable third generation architectures, Communications of the ACM 17 (1974) 412–421.

[75] VMware, Understanding full virtualization, paravirtualization, and hardware assist, White Paper, VMware Inc., 2007.

[76] M. Girola, A.M. Tarenzio, M. Lewis, M. Friedman, IBM Data Center Networking: Planning for Virtualization and Cloud Computing, first ed., IBM Redbooks Publication, 2011.

[77] K. Subramanian, Cloud advantage series: speed of deployment-cloud advantage series: speed of deployment, Online, 2009. Retrieved from: <http://www.cloudave.com/2523/cloud-advantage-series-speed-of-deployment/>.