Contents

CHAPTER 1

Introduction

The last decades have reinforced the idea that information processing can be done more efficiently centrally, on large farms of computing and storage systems accessible via the Internet. When computing resources in distant data centers are used rather than local computing systems, we talk about *network-centric computing* and *network-centric content*. Advancements in networking and other areas are responsible for the acceptance of the two new computing models and led to the *grid computing* movement in the early 1990s and, since 2005, to *utility computing* and *cloud computing*.

In *utility computing* the hardware and software resources are concentrated in large data centers and users can pay as they consume computing, storage, and communication resources. Utility computing often requires a cloud-like infrastructure, but its focus is on the business model for providing the computing services. *Cloud computing* is a path to utility computing embraced by major IT companies such as Amazon, Apple, Google, HP, IBM, Microsoft, Oracle, and others.

Cloud computing delivery models, deployment models, defining attributes, resources, and organization of the infrastructure discussed in this chapter are summarized in Figure 1.1. There are three cloud delivery models: *Software-as-a-Service* (SaaS), *Platform-as-a-Service* (PaaS), and *Infrastructure-as-a-Service* (IaaS), deployed as public, private, community, and hybrid clouds.

The defining attributes of the new philosophy for delivering computing services are as follows:

- Cloud computing uses Internet technologies to offer elastic services. The term *elastic computing* refers to the ability to dynamically acquire computing resources and support a variable workload. A cloud service provider maintains a massive infrastructure to support elastic services.
- The resources used for these services can be metered and the users can be charged only for the resources they use.
- Maintenance and security are ensured by service providers.
- Economy of scale allows service providers to operate more efficiently due to specialization and centralization.
- Cloud computing is cost-effective due to resource multiplexing; lower costs for the service provider are passed on to the cloud users.
- The application data is stored closer to the site where it is used in a device- and location-independent manner; potentially, this data storage strategy increases reliability and security and, at the same time, it lowers communication costs.

Cloud computing is a technical and social reality and an emerging technology. At this time, one can only speculate how the infrastructure for this new paradigm will evolve and what applications will migrate to it. The economical, social, ethical, and legal implications of this shift in technology, in which users rely on services provided by large data centers and store private data and software on systems they do not control, are likely to be significant.

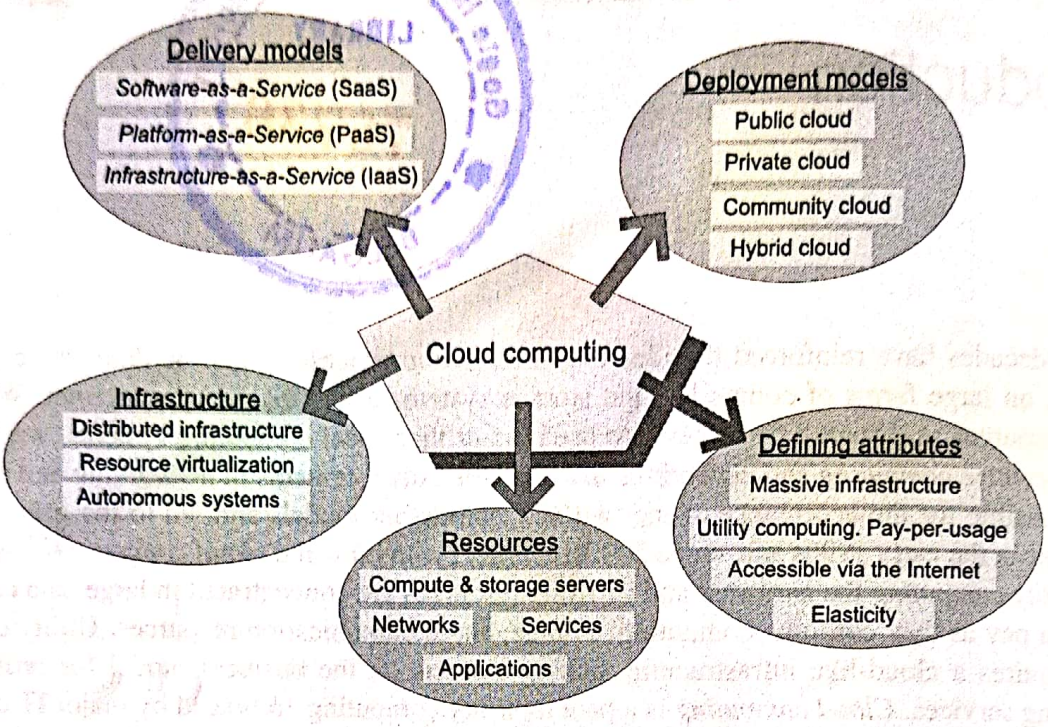

FIGURE 1.1

Cloud computing: Delivery models, deployment models, defining attributes, resources, and organization of the infrastructure.

Scientific and engineering applications, data mining, computational financing, gaming, and social networking as well as many other computational and data-intensive activities can benefit from cloud computing. A broad range of data, from the results of high-energy physics experiments to financial or enterprise management data to personal data such as photos, videos, and movies, can be stored on the cloud.

In early 2011 Apple announced the *iCloud,* a network-centric alternative for storing content such as music, videos, movies, and personal information; this content was previously confined to personal devices such as workstations, laptops, tablets, or smartphones. The obvious advantage of network-centric content is the accessibility of information from any site where users can connect to the Internet. Clearly, information stored on a cloud can be shared easily, but this approach raises major concerns: Is the information safe and secure? Is it accessible when we need it? Do we still own it?

In the next few years, the focus of cloud computing is expected to shift from building the infrastructure, today's main front of competition among the vendors, to the application domain. This shift in focus is reflected by Google's strategy to build a dedicated cloud for government organizations in the United States. The company states: "We recognize that government agencies have unique regulatory and compliance requirements for IT systems, and cloud computing is no exception. So we've invested a lot of time in understanding government's needs and how they relate to cloud computing."

In a discussion of technology trends, noted computer scientist Jim Gray emphasized that in 2003 the cost of communication in a wide area network had decreased dramatically and will continue to do so. Thus, it makes economical sense to store the data near the application [144] – in other words, to store

it in the cloud where the application runs. This insight leads us to believe that several new classes of cloud computing applications could emerge in the next few years [25].

As always, a good idea has generated a high level of excitement that translated into a flurry of publications – some of a scholarly depth, others with little merit or even bursting with misinformation. In this book we attempt to sift through the large volume of information and dissect the main ideas related to cloud computing. We first discuss applications of cloud computing and then analyze the infrastructure for the technology.

Several decades of research in parallel and distributed computing have paved the way for cloud computing. Through the years we have discovered the challenges posed by the implementation, as well as the algorithmic level, and the ways to address some of them and avoid the others. Thus, it is important to look back at the lessons we learned from this experience through the years; for this reason we start our discussion with an overview of parallel computing and distributed systems.

1.1 Network-centric computing and network-centric content

The concepts and technologies for network-centric computing and content evolved through the years and led to several large-scale distributed system developments:

- The Web and the semantic Web are expected to support composition of services (not necessarily computational services) available on the Web.[1]
- The Grid, initiated in the early 1990s by National Laboratories and Universities, is used primarily for applications in the area of science and engineering.
- Computer clouds, promoted since 2005 as a form of service-oriented computing by large IT companies, are used for enterprise computing, high-performance computing, Web hosting, and storage for network-centric content.

The need to share data from high-energy physics experiments motivated Sir Tim Berners-Lee, who worked at the European Organization for Nuclear Research (CERN) in the late 1980s, to put together the two major components of the World Wide Web: HyperText Markup Language (HTML) for data description and HyperText Transfer Protocol (HTTP) for data transfer. The Web opened a new era in data sharing and ultimately led to the concept of network-centric content.

The semantic Web[2] is an effort to enable laypeople to more easily find, share, and combine information available on the Web. In this vision, the information can be readily interpreted by machines, so machines can perform more of the tedious work involved in finding, combining, and acting upon information on the Web. Several technologies are necessary to provide a formal description of concepts, terms, and relationships within a given knowledge domain; they include the Resource Description Framework (RDF), a variety of data interchange formats, and notations such as RDF Schema (RDFS) and the Web Ontology Language (OWL).

[1] The Web is dominated by unstructured or semistructured data, whereas the semantic Web advocates inclusion of semantic content in Web pages.

[2] The term *semantic Web* was coined by Tim Berners-Lee to describe "a web of data that can be processed directly and indirectly by machines." It is a framework for data sharing among applications based on the Resource Description Framework (RDF). The semantic Web is "largely unrealized," according to Berners-Lee.

Gradually, the need to make computing more affordable and to liberate users from the concerns regarding system and software maintenance reinforced the idea of concentrating computing resources in data centers. Initially, these centers were specialized, each running a limited palette of software systems as well as applications developed by the users of these systems. In the early 1980s major research organizations such as the National Laboratories and large companies had powerful computing centers supporting large user populations scattered throughout wide geographic areas. Then the idea to link such centers in an infrastructure resembling the power grid was born; the model known as network-centric computing was taking shape.

A *computing grid* is a distributed system consisting of a large number of loosely coupled, heterogeneous, and geographically dispersed systems in different administrative domains. The term *computing grid* is a metaphor for accessing computer power with similar ease as we access power provided by the electric grid. Software libraries known as *middleware* have been furiously developed since the early 1990s to facilitate access to grid services.

The vision of the grid movement was to give a user the illusion of a very large virtual supercomputer. The autonomy of the individual systems and the fact that these systems were connected by wide-area networks with latency higher than the latency of the interconnection network of a supercomputer posed serious challenges to this vision. Nevertheless, several "Grand Challenge" problems, such as protein folding, financial modeling, earthquake simulation, and climate and weather modeling, run successfully on specialized grids. The Enabling Grids for Escience project is arguably the largest computing grid; along with the LHC Computing Grid (LCG), the Escience project aims to support the experiments using the Large Hadron Collider (LHC) at CERN which generate several gigabytes of data per second, or 10 PB (petabytes) per year.

In retrospect, two basic assumptions about the infrastructure prevented the grid movement from having the impact its supporters were hoping for. The first is the heterogeneity of the individual systems interconnected by the grid; the second is that systems in different administrative domains are expected to cooperate seamlessly. Indeed, the heterogeneity of the hardware and of system software poses significant challenges for application development and for application mobility. At the same time, critical areas of system management, including scheduling, optimization of resource allocation, load balancing, and fault tolerance, are extremely difficult in a heterogeneous system. The fact that resources are in different administrative domains further complicates many already difficult problems related to security and resource management. Although very popular in the science and engineering communities, the grid movement did not address the major concerns of the enterprise computing communities and did not make a noticeable impact on the IT industry.

Cloud computing is a technology largely viewed as the next big step in the development and deployment of an increasing number of distributed applications. The companies promoting cloud computing seem to have learned the most important lessons from the grid movement. Computer clouds are typically homogeneous. An entire cloud shares the same security, resource management, cost and other policies, and last but not least, it targets enterprise computing. These are some of the reasons that several agencies of the US Government, including Health and Human Services (HHS), the Centers for Disease Control (CDC), the National Aeronautics and Space Administration (NASA), the Navy's Next Generation Enterprise Network (NGEN), and the Defense Information Systems Agency (DISA), have launched cloud computing initiatives and conduct actual system development intended to improve the efficiency and effectiveness of their information processing needs.

The term *content* refers to any type or volume of media, be it static or dynamic, monolithic or modular, live or stored, produced by aggregation, or mixed. *Information* is the result of functions applied to content. The creation and consumption of audio and visual content are likely to transform the Internet to support increased quality in terms of resolution, frame rate, color depth, and stereoscopic information, and it seems reasonable to assume that the Future Internet[3] will be content-centric. The content should be treated as having meaningful semantic connotations rather than a string of bytes; the focus will be the information that can be extracted by content mining when users request named data and content providers publish data objects. Content-centric routing will allow users to fetch the desired data from the most suitable location in terms of network latency or download time. There are also some challenges, such as providing secure services for content manipulation, ensuring global rights management, control over unsuitable content, and reputation management.

Network-centric computing and network-centric content share a number of characteristics:

- Most applications are data-intensive. Computer simulation becomes a powerful tool for scientific research in virtually all areas of science, from physics, biology, and chemistry to archeology. Sophisticated tools for computer-aided design, such as Catia (Computer Aided Three-dimensional Interactive Application), are widely used in the aerospace and automotive industries. The widespread use of sensors contributes to increases in the volume of data. Multimedia applications are increasingly popular; the ever-larger media increase the load placed on storage, networking, and processing systems.
- Virtually all applications are network-intensive. Indeed, transferring large volumes of data requires high-bandwidth networks; parallel computing, computation steering,[4] and data streaming are examples of applications that can only run efficiently on low-latency networks.
- The systems are accessed using *thin clients* running on systems with limited resources. In June 2011 Google released Google Chrome OS, designed to run on primitive devices and based on the browser with the same name.
- The infrastructure supports some form of workflow management. Indeed, complex computational tasks require coordination of several applications; composition of services is a basic tenet of Web 2.0.

The advantages of network-centric computing and network-centric content paradigms are, at the same time, sources for concern; we discuss some of them:

- Computing and communication resources (CPU cycles, storage, network bandwidth) are shared and resources can be aggregated to support data-intensive applications. Multiplexing leads to a higher resource utilization; indeed, when multiple applications share a system, their peak demands for resources are not synchronized and the average system utilization increases. On the other hand, the management of large pools of resources poses new challenges as complex systems are subject to phase transitions. New resource management strategies, such as self-organization, and decisions based on approximate knowledge of the state of the system must be considered. Ensuring quality-of-service (QoS) guarantees is extremely challenging in such environments because total performance isolation is elusive.

[3]The term *Future Internet* is a generic concept referring to all research and development activities involved in the development of new architectures and protocols for the Internet.

[4]Computation steering in numerical simulation means to interactively guide a computational experiment toward a region of interest.

- Data sharing facilitates collaborative activities. Indeed, many applications in science, engineering, and industrial, financial, and governmental applications require multiple types of analysis of shared data sets and multiple decisions carried out by groups scattered around the globe. Open software development sites are another example of such collaborative activities. Data sharing poses not only security and privacy challenges but also requires mechanisms for access control by authorized users and for detailed logs of the history of data changes.
- Cost reduction. Concentration of resources creates the opportunity to pay as you go for computing and thus eliminates the initial investment and reduces significantly the maintenance and operation costs of the local computing infrastructure.
- User convenience and elasticity, that is the ability to accommodate workloads with very large peak-to-average ratios.

It is very hard to point out a single technological or architectural development that triggered the movement toward network-centric computing and network-centric content. This movement is the result of a cumulative effect of developments in microprocessor, storage, and networking technologies coupled with architectural advancements in all these areas and, last but not least, with advances in software systems, tools, programming languages, and algorithms to support distributed and parallel computing.

Through the years we have witnessed the breathtaking evolution of solid-state technologies which led to the development of multicore and many-core processors. Quad-core processors such as the AMD Phenom II X4, the Intel i3, i5, and i7 and hexa-core processors such as the AMD Phenom II X6 and Intel Core i7 Extreme Edition 980X are now used in the servers populating computer clouds. The proximity of multiple cores on the same die allows the cache coherency circuitry to operate at a much higher clock rate than would be possible if the signals were to travel off-chip.

Storage technology has also evolved dramatically. For example, solid-state disks such as RamSan-440 allow systems to manage very high transaction volumes and larger numbers of concurrent users. RamSan-440 uses DDR2 (double-data-rate) RAM to deliver 600,000 sustained random input/output operations per second (IOPS) and over 4 GB/s of sustained random `read` or `write` bandwidth, with latency of less than 15 microseconds, and it is available in 256 GB and 512 GB configurations. The price of memory has dropped significantly; at the time of this writing the price of a 1 GB module for a PC is approaching $10. Optical storage technologies and Flash memories are widely used nowadays.

The thinking in software engineering has also evolved and new models have emerged. The *three-tier model* is a software architecture and a software design pattern. The *presentation tier* is the topmost level of the application; typically, it runs on a desktop PC or workstation, uses a standard graphical user interface (GUI) and displays information related to services such as browsing merchandise, purchasing products, and managing shopping cart contents. The presentation tier communicates with other tiers by sending the results to the browser/client tier and all other tiers in the network. The *application/logic tier* controls the functionality of an application and may consist of one or more separate modules running on a workstation or application server; it may be multitiered itself, in which case the architecture is called an *n-tier architecture*. The *data tier* controls the servers where the information is stored; it runs a relational database management system (RDBMS) on a database server or a mainframe and contains the computer data storage logic. The data tier keeps data independent from application servers or processing logic and improves scalability and performance. Any of the tiers can be replaced independently; for example, a change of operating system in the presentation tier would only affect the user interface code.

1.2 Peer-to-peer systems

The distributed systems discussed in Chapter 2 allow access to resources in a tightly controlled environment. System administrators enforce security rules and control the allocation of physical rather than virtual resources. In all models of network-centric computing prior to utility computing, a user maintains direct control of the software and the data residing on remote systems.

This user-centric model, in place since the early 1960s, was challenged in the 1990s by the peer-to-peer (P2P) model. P2P systems can be regarded as one of the precursors of today's clouds. This new model for distributed computing promoted the idea of low-cost access to storage and central processing unit (CPU) cycles provided by participant systems; in this case, the resources are located in different administrative domains. Often the P2P systems are self-organizing and decentralized, whereas the servers in a cloud are in a single administrative domain and have a central management.

P2P systems exploit the network infrastructure to provide access to distributed computing resources. Decentralized applications developed in the 1980s, such as Simple Mail Transfer Protocol (SMTP), a protocol for email distribution, and Network News Transfer Protocol (NNTP), an application protocol for dissemination of news articles, are early examples of P2P systems. Systems developed in the late 1990s, such as the music-sharing system Napster, gave participants access to storage distributed over the network, while the first volunteer-based scientific computing, SETI@home, used free cycles of participating systems to carry out compute-intensive tasks.

The P2P model represents a significant departure from the client-server model, the cornerstone of distributed applications for several decades. P2P systems have several desirable properties [306]:

- They require a minimally dedicated infrastructure, since resources are contributed by the participating systems.
- They are highly decentralized.
- They are scalable; the individual nodes are not required to be aware of the global state.
- They are resilient to faults and attacks, since few of their elements are critical for the delivery of service and the abundance of resources can support a high degree of replication.
- Individual nodes do not require excessive network bandwidth the way servers used in case of the client-server model do.
- Last but not least, the systems are shielded from censorship due to the dynamic and often unstructured system architecture.

The undesirable properties of peer-to-peer systems are also notable: Decentralization raises the question of whether P2P systems can be managed effectively and provide the security required by various applications. The fact that they are shielded from censorship makes them a fertile ground for illegal activities, including distribution of copyrighted content.

In spite of its problems, the new paradigm was embraced by applications other than file sharing. Since 1999 new P2P applications such as the ubiquitous Skype, a Voice-over-Internet Protocol (VoIP) telephony service,[5] data-streaming applications such as Cool Streaming [386] and BBC's online video

[5] Skype allows close to 700 million registered users from many countries around the globe to communicate using a proprietary VoIP protocol. The system developed in 2003 by Niklas Zennström and Julius Friis was acquired by Microsoft in 2011 and nowadays is a hybrid P2P and client-server system.

service, content distribution networks such as CoDeeN [368], and volunteer computing applications based on the Berkeley Open Infrastructure for Networking Computing (BOINC) platform [21] have proved their appeal to users. For example, Skype reported in 2008 that 276 million registered Skype users have used more than 100 billion minutes for voice and video calls. The site `www.boinc.berkeley.edu` reports that at the end of June 2012 volunteer computing involved more than 275,000 individuals and more than 430,000 computers providing a monthly average of almost 6.3 petaFLOPS. It is also reported that peer-to-peer traffic accounts for a very large fraction of Internet traffic, with estimates ranging from 40% to more than 70%.

Many groups from industry and academia rushed to develop and test new ideas, taking advantage of the fact that P2P applications do not require a dedicated infrastructure. Applications such as *Chord* [334] and *Credence* [366] address issues critical to the effective operation of decentralized systems. *Chord* is a distributed lookup protocol to identify the node where a particular data item is stored. The routing tables are distributed and, whereas other algorithms for locating an object require the nodes to be aware of most of the nodes of the network, *Chord* maps a key related to an object to a node of the network using routing information about a few nodes only.

Credence is an object reputation and ranking scheme for large-scale P2P file-sharing systems. Reputation is of paramount importance for systems that often include many unreliable and malicious nodes. In the decentralized algorithm used by *Credence*, each client uses local information to evaluate the reputation of other nodes and shares its own assessment with its neighbors. The credibility of a node depends only on the votes it casts; each node computes the reputation of another node based solely on the degree of matching with its own votes and relies on like-minded peers. *Overcite* [337] is a P2P application to aggregate documents based on a three-tier design. The Web front-ends accept queries and display the results while servers crawl through the Web to generate indexes and to perform keyword searches; the Web back-ends store documents, meta-data, and coordination state on the participating systems.

The rapid acceptance of the new paradigm triggered the development of a new communication protocol allowing hosts at the network periphery to cope with the limited network bandwidth available to them. *BitTorrent* is a peer-to-peer file-sharing protocol that enables a node to download/upload large files from/to several hosts simultaneously.

The P2P systems differ in their architecture. Some do not have any centralized infrastructure, whereas others have a dedicated controller, but this controller is not involved in resource-intensive operations. For example, Skype has a central site to maintain user accounts; users sign in and pay for specific activities at this site. The controller for a BOINC platform maintains membership and is involved in task distribution to participating systems. The nodes with abundant resources in systems without any centralized infrastructure often act as *supernodes* and maintain information useful to increasing the system efficiency, such as indexes of the available content.

Regardless of the architecture, P2P systems are built around an *overlay network*, a virtual network superimposed over the real network. Methods to construct such an overlay, discussed in Section 7.10, consider a graph $G = (V, E)$, where V is the set of N vertices and E is the set of links between them.

Each node maintains a table of *overlay links* connecting it with other nodes of this virtual network, each node being identified by its IP address. Two types of overlay networks, unstructured and structured, are used by P2P systems. Random walks starting from a few bootstrap nodes are usually used by systems desiring to join an unstructured overlay. Each node of a structured overlay has a unique key that determines its position in the structure; the keys are selected to guarantee a uniform distribution in a

very large name space. Structured overlay networks use *key-based routing* (KBR); given a starting node v_0 and a key k, the function $KBR(v_0, k)$ returns the path in the graph from v_0 to the vertex with key k. Epidemic algorithms discussed in Section 7.12 are often used by unstructured overlays to disseminate network topology.

Module 1

1.3 Cloud computing: an old idea whose time has come

Once the technological elements were in place, it was only a matter of time until the economical advantages of cloud computing became apparent. Due to the economy of scale, large data centers – centers with more than 50,000 systems – are more economical to operate than medium-sized centers that have around 1,000 systems. Large data centers equipped with commodity computers experience a five to seven times decrease of resource consumption, including energy, compared to medium-sized centers [25]. The networking costs, in dollars per Mbit/s/month, are $95/13 = 7.1$ times larger, and the storage costs, in dollars per Gbyte/month, are $2.2/0.4 = 5.7$ times larger for medium-sized centers. Medium-sized centers have a larger administrative overhead – one system administrator for 140 systems versus one for 1,000 systems for large centers.

Data centers are very large consumers of electric energy to keep servers and the networking infrastructure running and for cooling. For example, there are 6,000 data centers in the United States and in 2006 they reportedly consumed 61 billion KWh, 1.5% of all electric energy in the U.S., at a cost of $4.5 billion. The power demanded by data centers was predicted to double from 2006 to 2011. Peak instantaneous demand was predicted to increase from 7 GW in 2006 to 12 GW in 2011, requiring the construction of 10 new power plants. In the United States the energy costs differ from state to state; for example 1 KWh costs 3.6 cents in Idaho, 10 cents in California, and 18 cents in Hawaii. Thus, data centers should be placed at sites with low energy cost.

The term *computer cloud* is overloaded, since it covers infrastructures of different sizes, with different management and different user populations. Several types of cloud are envisioned:

PCPH

- *Private cloud.* The infrastructure is operated solely for an organization. It may be managed by the organization or a third party and may exist on or off the premises of the organization.
- *Community cloud.* The infrastructure is shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be managed by the organizations or a third party and may exist on premises or off premises.
- *Public cloud.* The infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services.
- *Hybrid cloud.* The infrastructure is a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds).

A private cloud could provide the computing resources needed for a large organization, such as a research institution, a university, or a corporation. The argument that a private cloud does not support utility computing is based on the observation that an organization has to invest in the infrastructure and a user of a private cloud pays as it consumes resources [25]. Nevertheless, a private cloud could use the

same hardware infrastructure as a public one; its security requirements will be different from those for a public cloud and the software running on the cloud is likely to be restricted to a specific domain.

A natural question to ask is: Why could cloud computing be successful when other paradigms have failed? The reasons that cloud computing could be successful can be grouped into several general categories: technological advances, a realistic system model, user convenience, and financial advantages. A nonexhaustive list of reasons for the success of cloud computing includes these points:

- Cloud computing is in a better position to exploit recent advances in software, networking, storage, and processor technologies. Cloud computing is promoted by large IT companies where these new technological developments take place, and these companies have a vested interest in promoting the new technologies.
- A cloud consists of a homogeneous set of hardware and software resources in a single administrative domain. In this setup, security, resource management, fault tolerance, and quality of service are less challenging than in a heterogeneous environment with resources in multiple administrative domains.
- Cloud computing is focused on enterprise computing; its adoption by industrial organizations, financial institutions, healthcare organizations, and so on has a potentially huge impact on the economy.
- A cloud provides the illusion of infinite computing resources; its elasticity frees application designers from the confinement of a single system.
- A cloud eliminates the need for up-front financial commitment, and it is based on a pay-as-you-go approach. This has the potential to attract new applications and new users for existing applications, fomenting a new era of industrywide technological advancements.

In spite of the technological breakthroughs that have made cloud computing feasible, there are still major obstacles for this new technology; these obstacles provide opportunity for research. We list a few of the most obvious obstacles.

- *Availability of service.* What happens when the service provider cannot deliver? Can a large company such as General Motors move its IT to the cloud and have assurances that its activity will not be negatively affected by cloud overload? A partial answer to this question is provided by service-level agreements (SLAs).[6] A temporary fix with negative economical implications is *overprovisioning*, that is, having enough resources to satisfy the largest projected demand.
- *Vendor lock-in.* Once a customer is hooked to one provider, it is hard to move to another. The standardization efforts at National Institute of Standards and Technology (NIST) attempt to address this problem.
- *Data confidentiality and auditability.* This is indeed a serious problem; we analyze it in Chapter 9.
- *Data transfer bottlenecks.* Many applications are data-intensive. A very important strategy is to store the data as close as possible to the site where it is needed. Transferring 1 TB of data on a 1 Mbps network takes 8 million seconds, or about 10 days; it is faster and cheaper to use courier service and send data recoded on some media than to send it over the network. Very high-speed networks will alleviate this problem in the future; for example, a 1 Gbps network would reduce this time to 8,000 s, or slightly more than 2 h.

[6] SLAs are discussed in Section 3.8.

Performance unpredictability. This is one of the consequences of resource sharing. Strategies for performance isolation are discussed in Section 5.5.

Elasticity, the ability to scale up and down quickly. New algorithms for controlling resource allocation and workload placement are necessary. Autonomic computing based on self-organization and self-management seems to be a promising avenue.

There are other perennial problems with no clear solutions at this time, including software licensing and system bugs.

1.4 Cloud computing delivery models and services

According to the NIST reference model in Figure 1.2 [260], the entities involved in cloud computing are the *service consumer*, the entity that maintains a business relationship with and uses service from service providers; the *service provider*, the entity responsible for making a service available to service consumers; the *carrier*, the intermediary that provides connectivity and transport of cloud services between providers and consumers; the *broker*, an entity that manages the use, performance, and delivery of cloud services and negotiates relationships between providers and consumers; and the *auditor*, a party that can conduct independent assessment of cloud services, information system operations, performance, and security of the cloud implementation. An *audit* is a systematic evaluation of a cloud system that measures how well it conforms to a set of established criteria. For example, a security audit evaluates

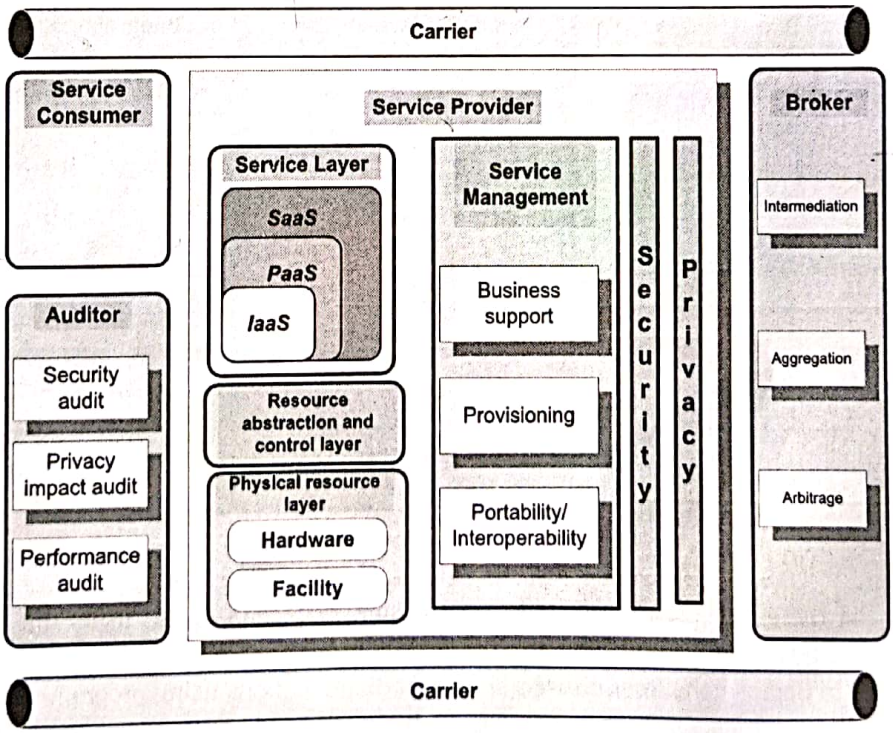

FIGURE 1.2

The entities involved in service-oriented computing and, in particular, in cloud computing, according to NIST. The carrier provides connectivity among service providers, service consumers, brokers, and auditors.

cloud security, a privacy-impact audit evaluates cloud privacy assurance, and a performance audit evaluates cloud performance.

We start with the observation that it is difficult to distinguish the services associated with cloud computing from those that any computer operations center would include [332]. Many of the services discussed in this section could be provided by a cloud architecture, but note that they are available in noncloud architectures as well.

Figure 1.3 presents the structure of the three delivery models, *SaaS*, *PaaS*, and *IaaS*, according to the Cloud Security Alliance [98].

Software-as-a-Service (SaaS) gives the capability to use applications supplied by the service provider in a cloud infrastructure. The applications are accessible from various client devices through a thin-client

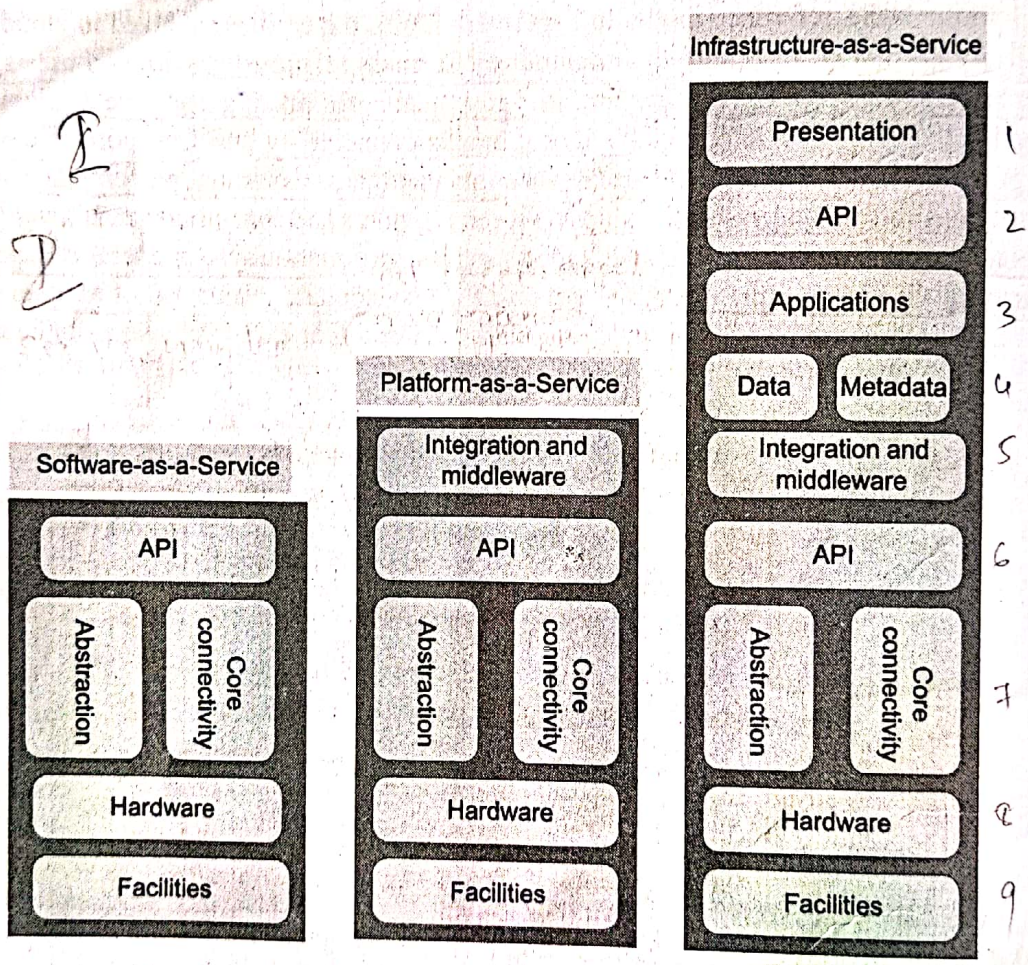

FIGURE 1.3

The structure of the three delivery models, *SaaS*, *PaaS*, and *IaaS*. *SaaS* gives users the capability to use applications supplied by the service provider but allows no control of the platform or the infrastructure. *PaaS* gives the capability to deploy consumer-created or acquired applications using programming languages and tools supported by the provider. *IaaS* allows the user to deploy and run arbitrary software, which can include operating systems and applications.

interface such as a Web browser (e.g., Web-based email). The user does not manage or control the underlying cloud infrastructure, including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings. Services offered include:

- Enterprise services such as workflow management, groupware and collaborative, supply chain, communications, digital signature, customer relationship management (CRM), desktop software, financial management, geo-spatial, and search [32].
- Web 2.0 applications such as metadata management, social networking, blogs, wiki services, and portal services.

The *SaaS* is not suitable for applications that require real-time response or those for which data is not allowed to be hosted externally. The most likely candidates for *SaaS* are applications for which:

- Many competitors use the same product, such as email.
- Periodically there is a significant peak in demand, such as billing and payroll.
- There is a need for Web or mobile access, such as mobile sales management software.
- There is only a short-term need, such as collaborative software for a project.

Platform-as-a-Service (PaaS) gives the capability to deploy consumer-created or acquired applications using programming languages and tools supported by the provider. The user does not manage or control the underlying cloud infrastructure, including network, servers, operating systems, or storage. The user has control over the deployed applications and, possibly, over the application hosting environment configurations. Such services include session management, device integration, sandboxes, instrumentation and testing, contents management, knowledge management, and Universal Description, Discovery, and Integration (UDDI), a platform-independent Extensible Markup Language (XML)-based registry providing a mechanism to register and locate Web service applications.

PaaS is not particulary useful when the application must be portable, when proprietary programming languages are used, or when the underlaying hardware and software must be customized to improve the performance of the application. The major *PaaS* application areas are in software development where multiple developers and users collaborate and the deployment and testing services should be automated.

Infrastructure-as-a-Service (IaaS) is the capability to provision processing, storage, networks, and other fundamental computing resources; the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly limited control of some networking components, such as host firewalls. Services offered by this delivery model include: server hosting, Web servers, storage, computing hardware, operating systems, virtual instances, load balancing, Internet access, and bandwidth provisioning.

The *IaaS* cloud computing delivery model has a number of characteristics, such as the fact that the resources are distributed and support dynamic scaling, it is based on a utility pricing model and variable cost, and the hardware is shared among multiple users. This cloud computing model is particulary useful when the demand is volatile and a new business needs computing resources and does not want to invest in a computing infrastructure or when an organization is expanding rapidly.

A number of activities are necessary to support the three delivery models; they include:

1. Service management and provisioning, including virtualization, service provisioning, call center, operations management, systems management, QoS management, billing and accounting, asset management, SLA management, technical support, and backups.
2. Security management, including ID and authentication, certification and accreditation, intrusion prevention, intrusion detection, virus protection, cryptography, physical security, incident response, access control, audit and trails, and firewalls.
3. Customer services such as customer assistance and online help, subscriptions, business intelligence, reporting, customer preferences, and personalization.
4. Integration services, including data management and development.

This list shows that a service-oriented architecture involves multiple subsystems and complex interactions among these subsystems. Individual subsystems can be layered; for example, in Figure 1.2 we see that the service layer sits on top of a resource abstraction layer, which controls the physical resource layer.

1.5 Ethical issues in cloud computing

Cloud computing is based on a paradigm shift with profound implications for computing ethics. The main elements of this shift are: (i) the control is relinquished to third-party services; (ii) the data is stored on multiple sites administered by several organizations; and (iii) multiple services interoperate across the network.

Unauthorized access, data corruption, infrastructure failure, and service unavailability are some of the risks related to relinquishing the control to third-party services; moreover, whenever a problem occurs, it is difficult to identify the source and the entity causing it. Systems can span the boundaries of multiple organizations and cross security borders, a process called *deperimeterization*. As a result of deperimeterization, "not only the border of the organization's IT infrastructure blurs, also the border of the accountability becomes less clear" [350].

The complex structure of cloud services can make it difficult to determine who is responsible in case something undesirable happens. In a complex chain of events or systems, many entities contribute to an action, with undesirable consequences. Some of them have the opportunity to prevent these consequences, and therefore no one can be held responsible – the so-called "problem of many hands."

Ubiquitous and unlimited data sharing and storage among organizations test the self-determination of information, the right or ability of individuals to exercise personal control over the collection, and use and disclosure of their personal data by others; this tests the confidence and trust in today's evolving information society. Identity fraud and theft are made possible by the unauthorized access to personal data in circulation and by new forms of dissemination through social networks, which could also pose a danger to cloud computing.

Cloud service providers have already collected petabytes of sensitive personal information stored in data centers around the world. The acceptance of cloud computing therefore will be determined by privacy issues addressed by these companies and the countries where the data centers are located. Privacy is affected by cultural differences; though some cultures favor privacy, other cultures emphasize community, and this leads to an ambivalent attitude toward privacy on the Internet, which is a global system.

The question of what can be done proactively about ethics of cloud computing does not have easy answers; many undesirable phenomena in cloud computing will only appear in time. However, the need for rules and regulations for the governance of cloud computing is obvious. The term *governance* means the manner in which something is governed or regulated, the method of management, or the system of regulations. Explicit attention to ethics must be paid by governmental organizations providing research funding for cloud computing; private companies are less constrained by ethics oversight and governance arrangements are more conducive to profit generation.

Accountability is a necessary ingredient of cloud computing; adequate information about how data is handled within the cloud and about allocation of responsibility are key elements for enforcing ethics rules in cloud computing. Recorded evidence allows us to assign responsibility; but there can be tension between privacy and accountability, and it is important to establish what is being recorded and who has access to the records.

Unwanted dependency on a cloud service provider, the so-called *vendor lock-in*, is a serious concern, and the current standardization efforts at NIST attempt to address this problem. Another concern for users is a future with only a handful of companies that dominate the market and dictate prices and policies.

1.6 Cloud vulnerabilities

Clouds are affected by malicious attacks and failures of the infrastructure (e.g., power failures). Such events can affect Internet domain name servers and prevent access to a cloud or can directly affect the clouds. For example, an attack at Akamai on June 15, 2004 caused a domain name outage and a major blackout that affected Google, Yahoo!, and many other sites. In May 2009 Google was the target of a serious denial-of-service (DoS) attack that took down services such Google News and Gmail for several days.

Lightning caused a prolonged downtime at Amazon on June 29 and 30, 2012; the *AWS* cloud in the Eastern region of the United States, which consists of 10 data centers across four availability zones, was initially troubled by utility power fluctuations, probably caused by an electrical storm. A June 29, 2012 storm on the East Coast took down some Virginia-based Amazon facilities and affected companies using systems exclusively in this region. *Instagram*, a photo-sharing service, was one of the victims of this outage, according to `http://mashable.com/2012/06/30/aws-instagram/`.

The recovery from the failure took a very long time and exposed a range of problems. For example, one of the 10 centers failed to switch to backup generators before exhausting the power that could be supplied by *uninterruptible power supply* (UPS) units. *AWS* uses "control planes" to allow users to switch to resources in a different region, and this software component also failed. The booting process was faulty and extended the time to restart *EC2* (*Elastic Computing*) and *EBS* (*Elastic Block Store*) services. Another critical problem was a bug in the elastic load balancer (ELB), which is used to route traffic to servers with available capacity. A similar bug affected the recovery process of the Relational Database Service (RDS). This event brought to light "hidden" problems that occur only under special circumstances.

A recent paper [126] identifies stability risks due to interacting services. A cloud application provider, a cloud storage provider, and a network provider could implement different policies, and the unpredictable interactions between load-balancing and other reactive mechanisms could lead to dynamic instabilities. The unintended coupling of independent controllers that manage the load, the power

consumption, and the elements of the infrastructure could lead to undesirable feedback and instability similar to the ones experienced by the policy-based routing in the Internet Border Gateway Protocol (BGP). For example, the load balancer of an application provider could interact with the power optimizer of the infrastructure provider. Some of these couplings may only manifest under extreme conditions and be very hard to detect under normal operating conditions, but they could have disastrous consequences when the system attempts to recover from a hard failure, as in the case of the *AWS* 2012 failure.

Clustering the resources in data centers located in different geographical areas is one of the means used today to lower the probability of catastrophic failures. This geographic dispersion of resources could have additional positive side effects; it can reduce communication traffic and energy costs by dispatching the computations to sites where the electric energy is cheaper, and it can improve performance by an intelligent and efficient load-balancing strategy. Sometimes a user has the option to decide where to run an application; we shall see in Section 3.1 that an *AWS* user has the option to choose the regions where the instances of his or her applications will run, as well as the regions of the storage sites. System objectives (e.g., maximize throughput, resource utilization, and financial benefits) have to be carefully balanced with user needs (e.g., low cost and response time and maximum availability).

The price to pay for any system optimization is increased system complexity, as we shall see in Section 10.7. For example, the latency of communication over a wide area network (WAN) is considerably larger than the one over a local area network (LAN) and requires the development of new algorithms for global decision making.

1.7 Major challenges faced by cloud computing

Cloud computing inherits some of the challenges of parallel and distributed computing discussed in Chapter 2; at the same time, it faces major challenges of its own. The specific challenges differ for the three cloud delivery models, but in all cases the difficulties are created by the very nature of utility computing, which is based on resource sharing and resource virtualization and requires a different trust model than the ubiquitous user-centric model we have been accustomed to for a very long time.

The most significant challenge is security [19]; gaining the trust of a large user base is critical for the future of cloud computing. It is unrealistic to expect that a public cloud will provide a suitable environment for all applications. Highly sensitive applications related to the management of the critical infrastructure, healthcare applications, and others will most likely be hosted by private clouds. Many real-time applications will probably still be confined to private clouds. Some applications may be best served by a hybrid cloud setup; such applications could keep sensitive data on a private cloud and use a public cloud for some of the processing.

The *SaaS* model faces similar challenges as other online services required to protect private information, such as financial or healthcare services. In this case a user interacts with cloud services through a well-defined interface; thus, in principle it is less challenging for the service provider to close some of the attack channels. Still, such services are vulnerable to DoS attack and the users are fearful of malicious insiders. Data in storage is most vulnerable to attack, so special attention should be devoted to the protection of storage servers. Data replication necessary to ensure continuity of service in case of storage system failure increases vulnerability. Data encryption may protect data in storage, but eventually data must be decrypted for processing, and then it is exposed to attack.

The *IaaS* model is by far the most challenging to defend against attacks. Indeed, an *IaaS* user has considerably more degrees of freedom than the other two cloud delivery models. An additional source of concern is that the considerable resources of a cloud could be used to initiate attacks against the network and the computing infrastructure.

Virtualization is a critical design option for this model, but it exposes the system to new sources of attack. The trusted computing base (TCB) of a virtual environment includes not only the hardware and the hypervisor but also the management operating system. As we shall see in Section 9.7, the entire state of a virtual machine (VM) can be saved to a file to allow migration and recovery, both highly desirable operations; yet this possibility challenges the strategies to bring the servers belonging to an organization to a desirable and stable state. Indeed, an infected VM can be inactive when the systems are cleaned up, and it can wake up later and infect other systems. This is another example of the deep intertwining of desirable and undesirable effects of basic cloud computing technologies.

The next major challenge is related to resource management on a cloud. Any systematic rather than ad hoc resource management strategy requires the existence of controllers tasked to implement several classes of policies: admission control, capacity allocation, load balancing, energy optimization, and last but not least, to provide QoS guarantees.

To implement these policies the controllers need accurate information about the global state of the system. Determining the state of a complex system with 10^6 servers or more, distributed over a large geographic area, is not feasible. Indeed, the external load, as well as the state of individual resources, changes very rapidly. Thus, controllers must be able to function with incomplete or approximate knowledge of the system state.

It seems reasonable to expect that such a complex system can only function based on self-management principles. But self-management and self-organization raise the bar for the implementation of logging and auditing procedures critical to the security and trust in a provider of cloud computing services. Under self-management it becomes next to impossible to identify the reasons that a certain action that resulted in a security breach was taken.

The last major challenge we want to address is related to interoperability and standardization. Vendor lock-in, the fact that a user is tied to a particular cloud service provider, is a major concern for cloud users (see Section 3.5). Standardization would support interoperability and thus alleviate some of the fears that a service critical for a large organization may not be available for an extended period of time. But imposing standards at a time when a technology is still evolving is not only challenging, it can be counterproductive because it may stifle innovation.

From this brief discussion the reader should realize the complexity of the problems posed by cloud computing and understand the wide range of technical and social problems cloud computing raises. If successful, the effort to migrate the IT activities of many government agencies to public and private clouds will have a lasting effect on cloud computing. Cloud computing can have a major impact on education, but we have seen little effort in this area.

1.8 Further reading

A very good starting point for understanding the major issues in cloud computing is the 2009 paper "Above the clouds: a Berkeley view of cloud computing" [25]. A comprehensive survey of peer-to-peer systems was published in 2010 [306]. Content distribution systems are discussed in [368]. The BOINC