CBCS SCHEME

USN | 1 | C | R | 2 | 2 | M | C | 0 | 8 | 4 |

22MCA332

Third Semester MCA Degree Examination, Dec.2023/Jan.2024
Cloud Computing

Time: 3 hrs. Max. Marks: 100

Note: 1. Answer any FIVE full questions, choosing ONE full question from each module.
2. M : Marks , L: Bloom's level , C: Course outcomes.

			M	L	C
		Module – 1			
Q.1	a.	Illustrate with figure bird's eye view of cloud computing and practical examples of cloud computing.	10	L2	CO1
	b.	Outline cloud reference model with definition, benefits and characteristics of cloud.	10	L2	CO1
		OR			
Q.2	a.	Categorize different distributed models supported in the growth of cloud computing.	10	L4	CO2
	b.	Explain following techniques: Virtualization, Web 2.0, Service-Oriented computing	10	L2	CO2
		Module – 2			
Q.3	a.	List and explain parallel and distributed computing differences.	06	L2	CO2
	b.	Explain SIMD and MISD hardware architectures with figures.	08	L2	CO2
	c.	Outline components of distributed computing along with neat figure.	06	L3	CO2
		OR			
Q.4	a.	Illustrate data centered and data flow architectures.	10	L3	CO2
	b.	Categorize and explain system architectural styles.	10	L2	CO2
		Module – 3			
Q.5	a.	Illustrate the characteristics of virtualized environments with neat diagrams.	08	L3	CO3
	b.	Explain full virtualization in hardware virtualization techniques.	06	L2	CO3
	c.	Explain application-level virtualization.	06	L2	CO3
		OR			
Q.6	a.	Explain hardware-level virtualization along with hypervisor functionalities.	10	L2	CO3
	b.	Discuss Xen example for para virtualization along with figure.	10	L2	CO3
		Module – 4			
Q.7	a.	Explain with figure the cloud reference architecture.	10	L2	CO4
	b.	Explain with figure IaaS reference implementation along with figure.	10	L2	CO4
		OR			
Q.8	a.	Classify and explain various types of cloud.	10	L2	CO4
	b.	Discuss the open challenges in cloud computing.	10	L2	CO4
		Module – 5			
Q.9	a.	List and explain compute services of Amazon web services.	10	L2	CO4
	b.	Explain Google AppEngine with platform architecture.	10	L2	CO4
		OR			
Q.10	a.	Illustrate with figure cloud environment for satellite data processing.	10	L3	CO4
	b.	Explain PropBox and iCloud applications with figure.	10	L2	CO4

* * * * *

Q1 a) Illustrate with figure bird's eye view of cloud computing and practical examples of cloud computing

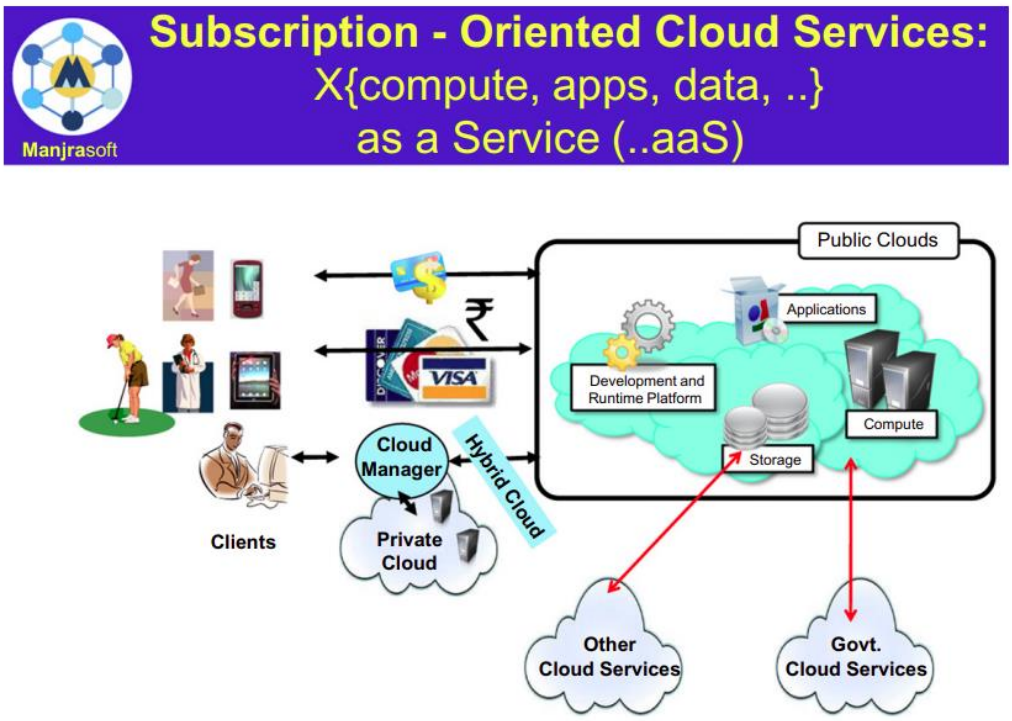


FIGURE 1.3
A bird's-eye view of cloud computing.

Cloud computing is helping enterprises, governments, public and private institutions, and research organizations shape more effective and demand-driven computing systems. Access to, as well as integration of, cloud computing resources and systems is now as easy as performing a credit card transaction over the Internet. Practical examples of such systems exist across all market segments:

• Large enterprises can offload some of their activities to cloud-based systems. Recently, the New York Times has converted its digital library of past editions into a Web-friendly format. This required a considerable amount of computing power for a short period of time. By renting Amazon EC2 and S3 Cloud resources, the Times performed this task in 36 hours and relinquished these resources, with no additional costs.

• Small enterprises and start-ups can afford to translate their ideas into business results more quickly, without excessive up-front costs. Animoto is a company that creates videos out of images, music, and video fragments submitted by users. The process involves a considerable amount of storage and backend processing required for producing the video, which is finally made available to the user. Animoto does not own a single server and bases its computing infrastructure entirely on Amazon Web Services, which are sized on demand according to the overall workload to be processed. Such workload can vary a lot and require instant scalability.3 Up-front investment is clearly not an effective solution for many companies, and cloud computing systems become an appropriate alternative.

• System developers can concentrate on the business logic rather than dealing with the complexity of infrastructure management and scalability. Little Fluffy Toys is a company in London that has developed a widget providing users with information about nearby bicycle rental services. The company has managed to back the widget's computing needs on Google AppEngine and be on the market in only one week.

• End users can have their documents accessible from everywhere and any device. Apple iCloud is a service that allows users to have their documents stored in the Cloud and access them from any device users connect to it. This makes it possible to take a picture while traveling with a smartphone, go back home and edit the same picture on your laptop, and have it show as updated on your tablet computer. This process is completely transparent to the user, who does not have to set up cables and connect these devices with each other.

Q1 b) Outline cloud refrence model with dfination benefits and charecteristics of cloud

A fundamental characteristic of cloud computing is the capability to deliver, on demand, a variety of IT services that are quite diverse from each other. This variety creates different perceptions of what cloud computing is among users. Despite this lack of uniformity, it is possible to classify cloud computing services offerings into three major categories: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS).

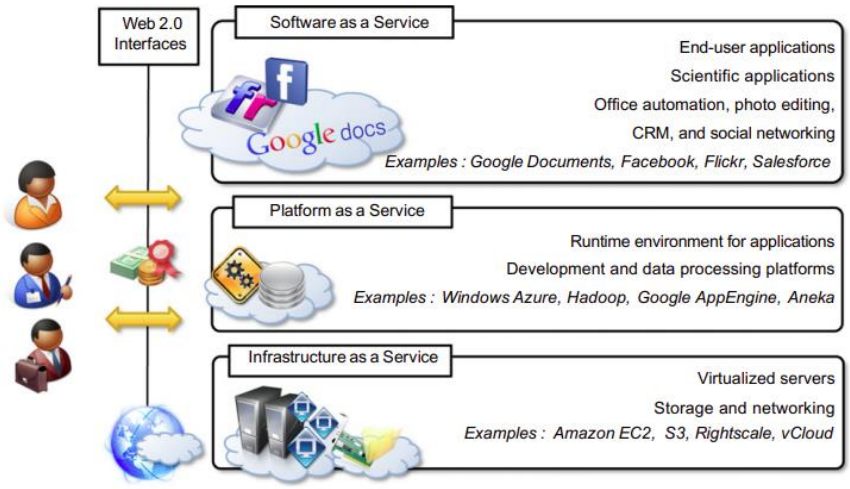


FIGURE 1.5
The Cloud Computing Reference Model.

At the base of the stack, **Infrastructure-as-a-Service** solutions deliver infrastructure on demand in the form of virtual hardware, storage, and networking. Virtual hardware is utilized to provide compute on demand in the form of virtual machine instances. These are created at users' request on the provider's infrastructure, and users are given tools and interfaces to configure the software stack installed in the virtual machine. The pricing model is usually defined in terms of dollars per hour, where the hourly cost is influenced by the characteristics of the virtual hardware. Virtual storage is delivered in the form of raw

disk space or object store.. Virtual networking identifies the collection of services that manage the networking among virtual instances and their connectivity to the Internet or private networks.

Platform-as-a-Service solutions are the next step in the stack. They deliver scalable and elastic runtime environments on demand and host the execution of applications. These services are backed by a core middleware platform that is responsible for creating the abstract environment where applications are deployed and executed. It is the responsibility of the service provider to provide scalability and to manage fault tolerance, while users are requested to focus on the logic of the application developed by leveraging the provider's APIs and libraries. This approach increases the level of abstraction at which cloud computing is leveraged but also constrains the user in a more controlled environment.

At the top of the stack, **Software-as-a-Service** solutions provide applications and services on demand. Most of the common functionalities of desktop applications—such as office automation, document management, photo editing, and customer relationship management (CRM) software—are replicated on the provider's infrastructure and made more scalable and accessible through a browser on demand. These applications are shared across multiple users whose interaction is isolated from the other users. The SaaS layer is also the area of social networking Websites, which leverage cloud-based infrastructures to sustain the load generated by their popularity.

Cloud computing has some interesting characteristics that bring benefits to both cloud service consumers (CSCs) and cloud service providers (CSPs). These characteristics are:

- No up-front commitments
- On-demand access
- Nice pricing
- Simplified application acceleration and scalability
- Efficient resource allocation
- Energy efficiency
- Seamless creation and use of third-party services

Q2 a) Categorize differnt distributed models supported in the growth of cloud computing

Three major milestones have led to cloud computing:

- Mainframe computing
- Cluster computing and

- Grid computing.

Mainframes. These were the first examples of large computational facilities leveraging multiple processing units. Mainframes were powerful, highly reliable computers specialized for large data movement and massive input/output (I/O) operations. They were mostly used by large organizations for bulk data processing tasks such as online transactions, enterprise resource planning, and other operations involving the processing of significant amounts of data.

Clusters. Cluster computing started as a low-cost alternative to the use of mainframes and supercomputers. The technology advancement that created faster and more powerful mainframes and supercomputers eventually generated an increased availability of cheap commodity machines as a side effect. These machines could then be connected by a high- bandwidth network and controlled by specific software tools that manage them as a single system. Starting in the 1980s. Cluster technology contributed considerably to the evolution of tools and frameworks for distributed computing, including Condor, Parallel Virtual Machine (PVM), and Message Passing Interface (MPI).

Grid computing appeared in the early 1990s as an evolution of cluster computing. In an analogy to the power grid, grid computing proposed a new approach to access large computational power, huge storage facilities, and a variety of services. A computing grid was a dynamic aggregation of heterogeneous computing nodes, and its scale was nationwide or even worldwide. Several developments made possible the diffusion of computing grids:

(a) Clusters became quite common resources;

(b) they were often underutilized;

(c) New problems were requiring computational power that went beyond the capability of single clusters; and

(d) The improvements in networking and the diffusion of the Internet made possible long- distance, high-bandwidth connectivity.

Q2 b) Explain following techniques: virtualization, web2.0, service-oriented computing

Virtualization

Virtualization is another core technology for cloud computing. It encompasses a collection of solutions allowing the abstraction of some of the fundamental elements for computing, such as hardware, runtime environments, storage, and networking. Virtualization has been around for more than 40 years, but its application has always been limited by technologies that did not allow an efficient use of virtualization solutions.

Virtualization is essentially a technology that allows creation of different computing environments. These environments are called virtual because they simulate the interface that is expected by a guest. The most common example of virtualization is hardware virtualization.

Virtualization technologies are also used to replicate runtime environments for programs. Applications in the case of process virtual machines (which include the foundation of technologies such as Java or .NET), instead of being executed by the operating system, are run by a specific program called a virtual machine. This technique allows isolating the execution of applications and providing a finer control on the resource they access.

Web 2.0

The Web is the primary interface through which cloud computing delivers its services. At present, the Web encompasses a set of technologies and services that facilitate interactive information sharing, collaboration, user-centered design, and application composition. This evolution has transformed the Web into a rich platform for application development and is known as Web 2.0. This term captures a new way in which developers architect applications and deliver services through the Internet and provides new experience for users of these applications and services.

Web 2.0 brings interactivity and flexibility into Web pages, providing enhanced user experience by gaining Web-based access to all the functions that are normally found in desktop applications. These capabilities are obtained by integrating a collection of standards and technologies such as XML, Asynchronous JavaScript and XML (AJAX), Web Services, and others. These technologies allow us to build applications leveraging the contribution of users, who now become providers of content.

Web 2.0 applications are extremely dynamic: they improve continuously, and new updates and features are integrated at a constant rate by following the usage trend of the community. There is no need to deploy new software releases on the installed base at the client side.

Service-oriented computing

Service orientation is the core reference model for cloud computing systems. This approach adopts the concept of services as the main building blocks of application and system development.

Service-oriented computing (SOC) supports the development of rapid, low-cost, flexible, interoperable, and evolvable applications and systems.

A service is an abstraction representing a self-describing and platform-agnostic component that can perform any function—anything from a simple function to a complex business process.

A service is supposed to be loosely coupled, reusable, programming language independent, and location transparent. Loose coupling allows services to serve different scenarios more easily and makes them reusable. Independence from a specific platform increases services accessibility. Thus, a wider range of clients, which can look up services in global registries and consume them in a location-transparent manner, can be served.

Service-oriented computing introduces and diffuses two important concepts, which are also fundamental to cloud computing: quality of service (QoS) and Software-as-a-Service (SaaS).

- Quality of service (QoS) identifies a set of functional and nonfunctional attributes that can be used to evaluate the behavior of a service from different perspectives. These could be performance metrics such as response time, or security attributes, transactional integrity, reliability, scalability, and availability.
- The concept of Software-as-a-Service introduces a new delivery model for applications. The term has been inherited from the world of application service providers (ASPs), which deliver software services-based solutions across the wide area network from a central datacenter and make them available on a subscription or rental basis.

Q3 a) List and explain parallel and distributed differences

The terms parallel computing and distributed computing are often used interchangeably, even though they mean slightly different things.

Parallel implies a tightly coupled system. Parallel computing refers to a model in which the computation is divided among several processors sharing the same memory. The architecture of a parallel computing system is often characterized by the homogeneity of components: each processor is of the same type, and it has the same capability as the others. The shared memory has a single address space, which is accessible to all the processors. Parallel programs are then broken down into several units of execution that can be allocated to different processors and can communicate with each other by means of the shared memory.

Distributed refers to a wider class of system, including those that are tightly coupled. The term distributed computing encompasses any architecture or system that allows the computation to be broken down into units and executed concurrently on different computing elements, whether these are processors on different nodes, processors on the same computer, or cores within the same processor. Therefore, distributed computing includes a wider range of systems and applications than parallel computing.

Q3 b) Explain SMID and MISD hardware architecture with figures

The core elements of parallel processing are CPUs. Based on the number of instruction and data streams that can be processed simultaneously, computing systems are classified into the following four categories:

• Single-instruction, single-data (SISD) systems

• Single-instruction, multiple-data (SIMD) systems

• Multiple-instruction, single-data (MISD) systems

• Multiple-instruction, multiple-data (MIMD) systems

systems are IBM PC, Macintosh, and workstations

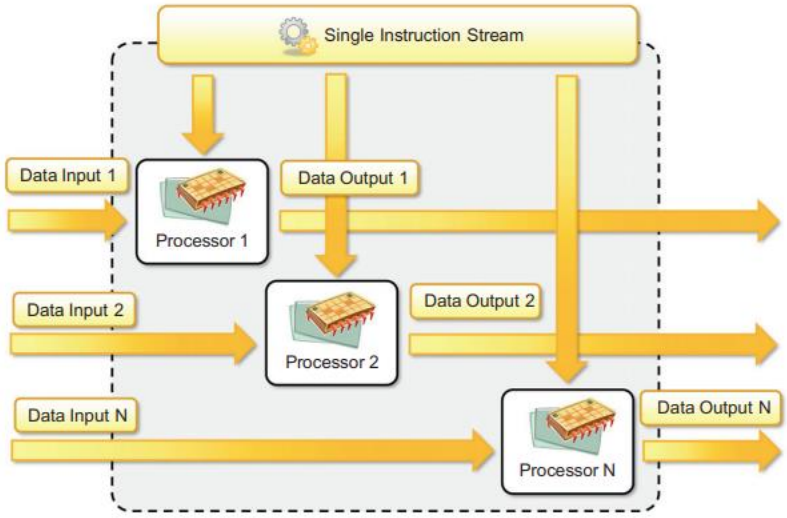
Single-instruction, multiple-data (SIMD) systems

An SIMD computing system is a multiprocessor machine capable of executing the same instruction on all the CPUs but operating on different data streams. Machines based on an SIMD model are well suited to scientific computing since they involve lots of vector and matrix operations. For instance, statements such as

Ci 5 Ai Bi

can be passed to all the processing elements (PEs); organized data elements of vectors A and B can be divided into multiple sets (N-sets for N PE systems); and each PE can process one data set. Dominant representative SIMD systems are Cray's vector processing machine and Thinking Machines' cm*.

$$y = \sin(x) + \cos(x) + \tan(x)$$



Multiple-instruction, single-data (MISD) systems

An MISD computing system is a multiprocessor machine capable of executing different instructions on different PEs but all of them operating on the same data set (see Figure 2.4). For instance, statements such as

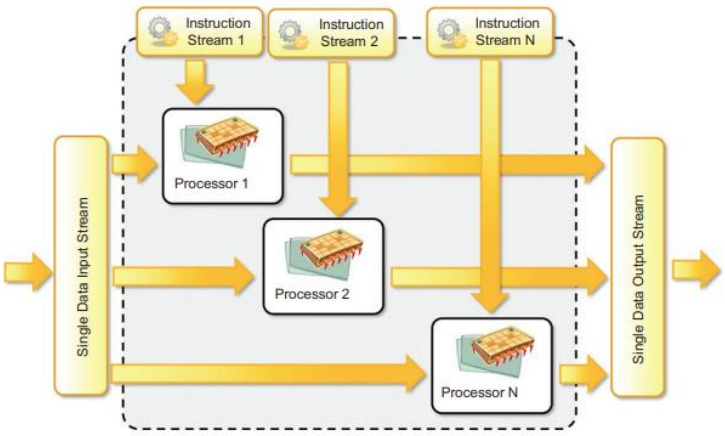


FIGURE 2.4
Multiple-instruction, single-data (MISD) architecture.

perform different operations on the same data set. Machines built using the MISD model are not useful in most of the applications; a few machines are built, but none of them are available commercially. They became more of an intellectual exercise than a practical configuration.

Q3 c) Outline components of distribted computing along with neat figure

A distributed system is the result of the interaction of several components that traverse the entire computing stack from hardware to software.

It emerges from the collaboration of several elements that by working together give users the illusion of a single coherent system.

Figure 2.10 provides an overview of the different layers that are involved in providing the services of a distributed system.

At the very **bottom layer**, computer and network hardware constitute the physical infrastructure; these components are directly managed by the next layer, operating system, which provides the basic services for interprocess communication (IPC), process scheduling and management, and resource management in terms of file system and local devices. Taken together these two layers become the platform on top of which specialized software is deployed to turn a set of networked computers into a distributed system. At the operating system level, IPC services are implemented on top of standardized communication protocols such Transmission Control Protocol/Internet Protocol (TCP/IP), User Datagram Protocol (UDP) or others.

The **middleware layer** leverages such services to build a uniform environment for the development and deployment of distributed applications. This layer supports the programming paradigms for distributed systems the middleware develops its own protocols, data formats, and programming language or frameworks for the development of distributed applications. All of them constitute a uniform interface to distributed application developers that is completely independent from the underlying operating system and hides all the heterogeneities of the bottom layers.

The **top layer** of the distributed system stack is represented by the applications and services designed and developed to use the middleware. These can serve several purposes and often expose their features in the form of graphical user interfaces (GUIs) accessible locally or through the Internet via a Web browser.

A very good example is constituted by Infrastructure-as-a-Service (IaaS) providers such as Amazon Web Services

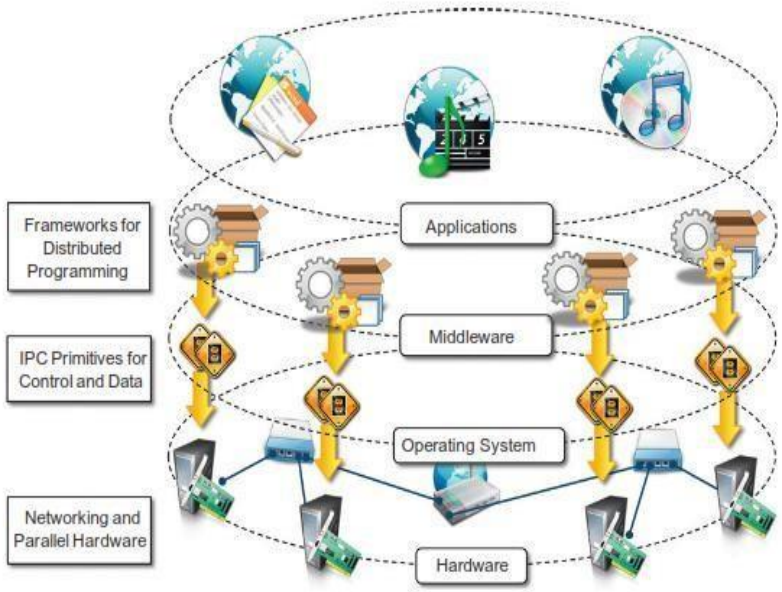


FIGURE 2.10
A layered view of a distributed system.

(AWS), which provide facilities for creating virtual machines, organizing them together into a cluster, and deploying applications and systems on top.

Q4 a) Illustrate data centered and data flow architectres

Data centered architectures

These architectures identify the data as the fundamental element of the software system, and access to shared data is the core characteristic of the data-centered architectures. Therefore, especially within the context of distributed and parallel computing systems, integrity of data is the overall goal for such systems. The repository architectural style is the most relevant reference model in this category. It is characterized by two main components: the central data structure, which represents the current state of the system, and a collection of independent components, which operate on the central data. The ways in which the independent components interact with the central data structure can be very heterogeneous. In particular, repository-based architectures differentiate and specialize further into subcategories according to the choice of control discipline to apply for the shared data structure. Of particular interest are databases and blackboard systems. In the former group the dynamic of the system is controlled by the independent components, which, by issuing an operation on the central repository, trigger the

selection of specific processes that operate on data. In blackboard systems, the central data structure is the main trigger for selecting the processes to execute.

The blackboard architectural style is characterized by three main components:

• Knowledge sources. These are the entities that update the knowledge base that is maintained in the blackboard.

• Blackboard. This represents the data structure that is shared among the knowledge sources and stores the knowledge base of the application.

• Control. The control is the collection of triggers and procedures that govern the interaction with the blackboard and update the status of the knowledge base.

Within this reference scenario, knowledge sources, which represent the intelligent agents sharing the blackboard, react opportunistically to changes in the knowledge base, almost in the same way that a group of specialists brainstorm in a room in front of a blackboard. Blackboard models have become popular and widely used for artificial intelligent applications in which the blackboard maintains the knowledge about a domain in the form of assertions and rules, which are entered by domain experts. These operate through a control shell that controls the problem-solving activity of the system. Particular and successful applications of this model can be found in the domains of speech recognition and signal processing.

Data flow architectures

- Data Flow Architecture is transformed input data by a **series of computational** ormanipulative components into output data.

There are Two types of execution sequences between modules:

1. Batch Sequential style
2. Pipe and Filter style

1. Batch Sequential style

The batch sequential style is characterized by an ordered sequence of separate programsexecuting one after the other.

These programs are chained together by providing as input for the next program the output generated by the last program after its completion, which is most likely in the form of a file.

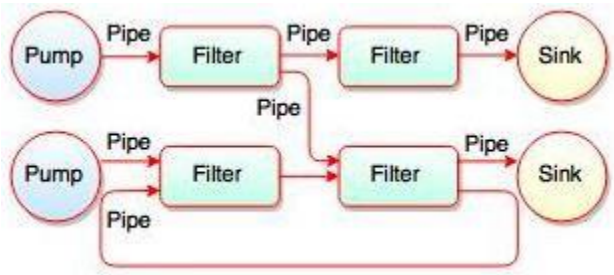
The above diagram shows the flow of batch sequential architecture. It provides simpler divisions on subsystems and each subsystem can be an independent program working on input data and produces output data.

For example, many distributed applications for scientific computing are defined by jobs expressed as sequence of programs that, for **Example**, pre-filter, analyse, and post process data. It is very common to compose these phases using the batch sequential style.

2. Pipe and Filter style

What is meant by Pipe?

- Pipe is a connector which passes the data from one filter to the next.
- Pipe is a directional sequence of data transformations of data implemented by a databuffer to store all data, until the next filter has time to process it.
- It transfers the data from one data source to one data sink.
- Pipes are the stateless data stream.



What are the Filters?

- Filter is a component.
- It has interfaces from which a set of inputs can flow in, and a set of outputs can flow out.
- It transforms and refines input data.
- Filters are the independent entities.

Q4 b) Categorize and explain system archtictureal styles

System architectural styles cover the physical organization of components and processes over adistributed infrastructure.

There are Two fundamental reference style

1. Client/Server and
2. Peer-to-Peer

1. Client / Server

The client/server model features two major components: a server and a client. These two components interact with each other through a network connection using a given protocol. The communication is unidirectional: The client issues a request to the server, and after processing the request the server returns a response. There could be multiple client components issuing requests to a server that is passively waiting for them. Hence, the important operations in the client- server paradigm are request, accept (client side), and listen and response (server side).

For the client design, we identify two major models:

a. Thin-client model.
b. Fat-client model.

Thin-client model. In this model, the load of data processing and transformation is put on the server side, and the client has a light implementation that is mostly concerned with retrieving and returning the data it is being asked for, with no considerable further processing.

a. Fat-client model. In this model, the client component is also responsible for processing and transforming the data before returning it to the user, whereas the server features a relatively light implementation that is mostly concerned with the management of access to the data.

1) Two-tier architecture

This architecture partitions the systems into two tiers, which are located one in the client component and the other

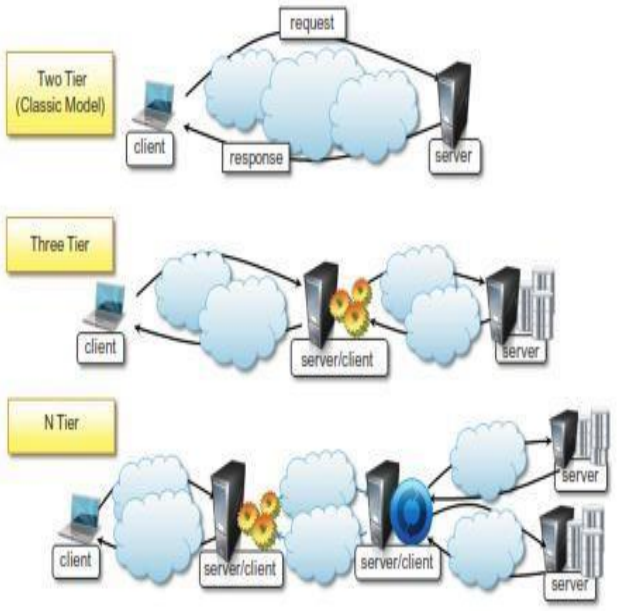


FIGURE 2.12
Client/server architectural styles.

on the server. The client is responsible for the presentation tier by providing a user interface; the server concentrates the application logic and the data store into a single tier. The server component is generally deployed on a powerful machine that is capable of processing user requests, accessing data, and executing the application logic to provide a client with a response. This architecture is suitable for systems of limited size and suffers from scalability issues.

2) Three-tier architecture/N-tier architecture

The three-tier architecture separates the presentation of data, the application logic, and the data storage into three tiers. This architecture is generalized into an N-tier model in case it is necessary to further divide the stages composing the application logic and storage tiers. This model is generally more scalable than the two-tier one because it is possible to distribute the tiers into several computing nodes, thus isolating the performance bottlenecks.

2. Peer-to-Peer

The peer-to-peer model, depicted in Figure 2.13, introduces a symmetric architecture in which all the components called peers, play the same role, and incorporate both client and server capabilities of the client/server model.

Each peer acts as a server when it processes requests from other peers and as a client when it issues requests to other peers.

With respect to the client/server model that partitions the responsibilities of the IPC between server and clients, the peer-to peer model attributes the same responsibilities to each component. Therefore, this model is quite suitable for highly decentralized architecture, which can scale better along the dimension of the number of peers.

FIGURE 2.13
Peer-to-peer architectural style.

Q5 a) Illustrate the characterisitcs of virtualized environment with neat diagram.

- Increased performance and computing capacity

Now a days the average end-user desktop PC is powerful enough to fulfil almost all the needs of everyday computing and there is an extra capacity that is rarely used.

Individual PCs have resources enough to host a virtual machine manager and execute a virtual machine with by far acceptable performance

Likewise, the supercomputers can provide immense compute power that can accommodate the execution of hundreds or thousands of virtual machines

- Underutilized hardware and software resources

Computers today are so powerful that in most cases only a fraction of their capacity is used by an application or the system

To improve the efficiency of the IT infrastructure, to transparently provide such a service, it would be necessary to deploy a completely separate environment, which can be achieved through virtualization.

- Lack of space

The continuous need for additional capacity, whether storage or compute power, makes data centers grow quickly adding the size of data centers for every need by the IT enterprise would be infeasible

This condition, along with hardware underutilization, has led to the diffusion of a technique called server consolidation, for which virtualization technologies are fundamental

- Greening initiatives

Maintaining a data center operation not only involves keeping servers on, but a great deal of energy is also consumed in keeping them cool.

Infrastructures for cooling have a significant impact on the carbon footprint of a data center

Virtualization technologies can provide an efficient way of consolidating servers thereby reducing the power consumption and hence the carbon footprint

- Rise of administrative costs

Power consumption and cooling costs have now become higher than the cost of IT equipment

the higher the number of servers that have to be managed, the higher the administrative costs

Virtualization can help reduce the number of required servers for a given workload, thus reducing the cost of the administrative personnel

Q5 b) Explain full virtualization in hardware virtualization techniques

Full virtualization refers to the ability to run a program, most likely an operating system, directly on top of a virtual machine and without any modification, as though it were run on the raw hardware. To make this possible, virtual machine managers are required to provide a complete emulation of the entire underlying hardware. The principal advantage of full virtualization is complete isolation, which leads to enhanced security, ease of emulation of different architectures, and coexistence of different systems on the same platform. Whereas it is a desired goal for many virtualization solutions, full virtualization poses important concerns related to performance and technical implementation. A key challenge is the interception of privileged instructions such as I/O instructions: Since they change the state of the resources exposed by the host, they have to be contained within the virtual machine manager. A simple solution to achieve full virtualization is to provide a virtual environment for all the instructions, thus posing some limits on performance. A successful and efficient implementation of full virtualization is obtained with a combination of hardware and software, not allowing potentially harmful instructions to be executed directly on the host. This is what is accomplished through hardware-assisted virtualization.

Q5 c) Explain application level virtualization

- Application-level virtualization is a technique allowing applications to be run in runtime environments that do not natively support all the features required by such applications. In this scenario, applications are not installed in the expected runtime environment but are run as though they were.
- Emulation can also be used to execute program binaries compiled for different hardware architectures. In this case, one of the following strategies can be implemented:
- Interpretation. In this technique every source instruction is interpreted by an emulator for executing native ISA instructions, leading to poor performance. Interpretation has a minimal start-up cost but a huge overhead, since each instruction is emulated.
- Binary translation. In this technique every source instruction is converted to native instructions with equivalent functions. After a block of instructions is translated, it is cached and reused. Binary translation has a large initial overhead cost, but over time it is subject to better performance, since previously translated instruction blocks are directly executed.
- Application virtualization is a good solution in the case of missing libraries in the host operating system; in this case a replacement library can be linked with the application, or library calls can be remapped to existing functions available in the host system. Another advantage is that in this case the virtual machine manager is much lighter since it provides a partial emulation of the runtime environment compared to hardware virtualization.

- One of the most popular solutions implementing application virtualization is Wine, which is a software application allowing Unix-like operating systems to execute programs written for the Microsoft Windows platform. Wine features a software application acting as a container for the guest application and a set of libraries, called Winelib, that developers can use to compile applications to be ported on Unix systems.
- Wine takes its inspiration from a similar product from Sun, Windows Application Binary Interface (WABI), which implements the Win 16 API specifications on Solaris. A similar solution for the Mac OS X environment is CrossOver, which allows running Windows applications directly on the Mac OS X operating system. VMware ThinApp, another product in this area, allows capturing the setup of an installed application and packaging it into an executable image isolated from the hosting operating system
-

Q6 a) Explain hardware-level virtualization along with hypervisor functionalities.

- It provides an abstract execution environment in terms of **computer hardware** on **top** of which a guest **operating system** can be run.
- The guest is represented by the operating system, the host by the physical computer hardware, the virtual machine by its emulation, and the virtual machine manager by the **hypervisor** (see Figure 3.6).
- The hypervisor is generally a program or a combination of software and hardware that allows the abstraction of the underlying physical hardware.
- **Hardware-level virtualization is also called system virtualization**, since it provides **ISA** to **virtual machines**, which is the representation of the hardware interface of a system.

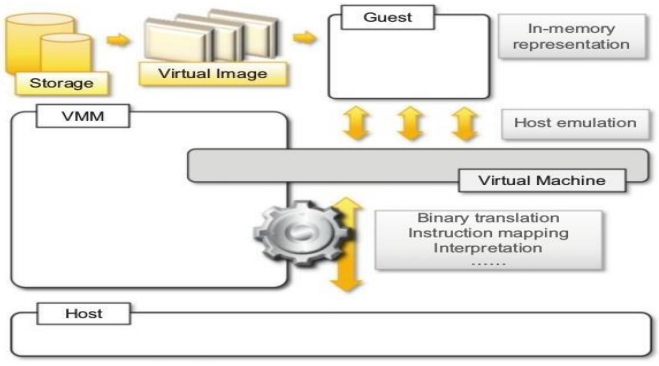


FIGURE 3.6
A hardware virtualization reference model.

A fundamental element of hardware virtualization is the hypervisor, or virtual machine manager (VMM). It recreates a hardware environment in which guest operating systems are installed. There are two major types of hypervisor: Type I and Type II (see Figure 3.7).

• Type I hypervisors run directly on top of the hardware. Therefore, they take the place of the operating systems and interact directly with the ISA interface exposed by the underlying hardware, and they emulate this interface in order to allow the management of guest operating systems. This type of hypervisor is also called a native virtual machine since it runs natively on hardware

• Type II hypervisors require the support of an operating system to provide virtualization services. This means that they are programs managed by the operating system, which interact with it through the ABI and emulate the ISA of virtual hardware for guest operating systems. This type of hypervisor is also called a hosted virtual machine since it is hosted within an operating system.

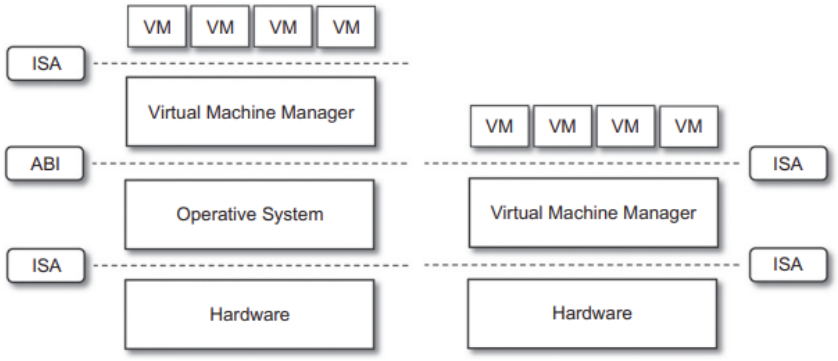


FIGURE 3.7
Hosted (left) and native (right) virtual machines. This figure provides a graphical representation of the two types of hypervisors.

Conceptually, a virtual machine manager is internally organized as described in Figure 3.8. Three main modules, dispatcher, allocator, and interpreter, coordinate their activity in order to emulate the underlying hardware. The dispatcher constitutes the entry point of the monitor and reroutes the instructions issued by the virtual machine instance to one of the two other modules. The allocator is responsible for deciding the system resources to be provided to the VM: whenever a virtual machine tries to execute an instruction that results in changing the machine resources associated with that VM, the allocator is invoked by the dispatcher. The interpreter module consists of interpreter routines. These are executed whenever a virtual machine executes a privileged instruction: a trap is triggered and the corresponding routine is executed. The design and architecture of a virtual machine manager, together with the underlying hardware design of the host machine, determine the full realization of hardware virtualization, where a guest operating system can be transparently executed on top of a VMM as though it were run on the underlying hardware. The criteria that need to be met by a virtual machine manager to efficiently support virtualization were established by Goldberg and Popek in 1974 [23]. Three properties have to be satisfied:

• Equivalence. A guest running under the control of a virtual machine manager should exhibit the same behavior as when it is executed directly on the physical host.

• Resource control. The virtual machine manager should be in complete control of virtualized resources.

• Efficiency. A statistically dominant fraction of the machine instructions should be executed without intervention from the virtual machine manager.

Q6 b) Discuss Xen example for para virtualization along with figure.

Xen is an open-source initiative implementing a virtualization platform based on paravirtualization. Initially developed by a group of researchers at the University of Cambridge in the United Kingdom, Xen now has a large open-source community backing it. Citrix also offers it as a commercial solution, XenSource. Xen-based technology is used for either desktop virtualization or server virtualization, and recently it has also been used to provide cloud computing solutions by means of Xen Cloud Platform (XCP). At the basis of all these solutions is the Xen Hypervisor, which constitutes the core technology of Xen. Recently Xen has been advanced to support full virtualization using hardware-assisted virtualization. Xen is the most popular implementation of paravirtualization, which, in contrast with full virtualization, allows high-performance execution of guest operating systems. This is made possible by eliminating the performance loss while executing instructions that require special management. This is done by modifying portions of the guest operating systems run by Xen with reference to the execution of such instructions. Therefore it is not a transparent solution for implementing virtualization. This is particularly true for x86, which is the most popular architecture on commodity machines and servers.

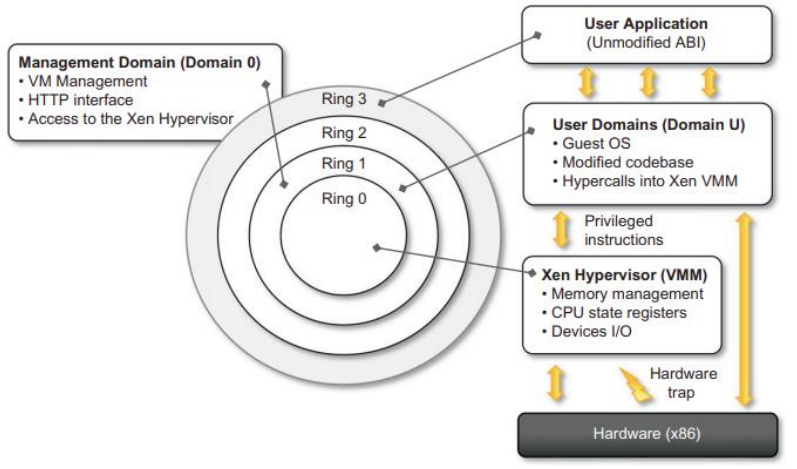


FIGURE 3.11
Xen architecture and guest OS management.

Figure 3.11 describes the architecture of Xen and its mapping onto a classic x86 privilege model. A Xen-based system is managed by the Xen hypervisor, which runs in the highest privileged mode and controls the access of guest operating system to the underlying hardware. Guest operating systems are executed

within domains, which represent virtual machine instances. Moreover, specific control software, which has privileged access to the host and controls all the other guest operating systems, is executed in a special domain called Domain 0. This is the first one that is loaded once the virtual machine manager has completely booted, and it hosts a HyperText Transfer Protocol (HTTP) server that serves requests for virtual machine creation, configuration, and termination. This component constitutes the embryonic version of a distributed virtual machine manager, which is an essential component of cloud computing systems providing Infrastructure-as-a-Service (IaaS) solutions.

Popek and Goldberg provided a classification of the instruction set and proposed three theorems that define the properties that hardware instructions need to satisfy in order to efficiently support virtualization

THEOREM 3.1 For any conventional third-generation computer, a VMM may be constructed if the set of sensitive instructions for that computer is a subset of the set of privileged instructions.

THEOREM 3.2 A conventional third-generation computer is recursively virtualizable if:

• It is virtualizable and

• A VMM without any timing dependencies can be constructed for it.

THEOREM 3.3 A hybrid VMM may be constructed for any conventional third-generation machine in which the set of user-sensitive instructions is a subset of the set of privileged instructions.

Q7 a) Explain with figure the cloud reference architecture

It is possible to organize all the concrete realizations of cloud computing into a layered view covering the entire stack (see Figure 4.1), from hardware appliances to software systems. Cloud infrastructure can be heterogeneous in nature because a variety of resources, such as clusters and even networked PCs, can be used to build it.

- The physical infrastructure is managed by the core middleware, the objectives of which are to provide an appropriate runtime environment for applications and to best utilize resources.
- At the bottom of the stack, virtualization technologies are used to guarantee runtime environment customization, application isolation, sandboxing, and quality of service. Hardware virtualization is most used at this level. Hypervisors manage the pool of resources and expose the distributed infrastructure as a collection of virtual machines. By using virtual machine technology, it is possible tofinely partition the hardware resources such as CPU and memory and to virtualize specific devices,thus meeting the requirementsof users and applications. This solution is generally paired with storage and network virtualization strategies, which allow the infrastructure to be completely virtualized and controlled.
- Infrastructure management is the key function of core middleware, which supports capabilities such as negotiation of the quality of service, admission control, execution management and

monitoring, accounting, and billing. The combination of cloud hosting platforms and resources is generally classified as an Infrastructure-as-a-Service(IaaS)solution.

- We can organize the different examples of IaaS into two categories: Some of them provide both the management layer and the physical infrastructure; others provide only the management layer (IaaS (M)).
- User-level middleware: The range of tools include Web-based interfaces, command-line tools, and frameworks for concurrent and distributed programming. In this scenario, users develop their applications specifically for the cloud by using the API exposed at the user- level middleware. For this reason, this approach is also known as Platform-as-a-Service (PaaS) because the service offered to the user is a development platform rather than an infrastructure.
- The top layer of the reference model is User Application level: These are referred as Software-as-a-Service (SaaS). In most cases these are Web-based applications that rely on the cloud to provide service to end users.
- The horsepower of the cloud provided by IaaS and PaaS solutions allows independent software vendors to deliver their application services over the Internet.

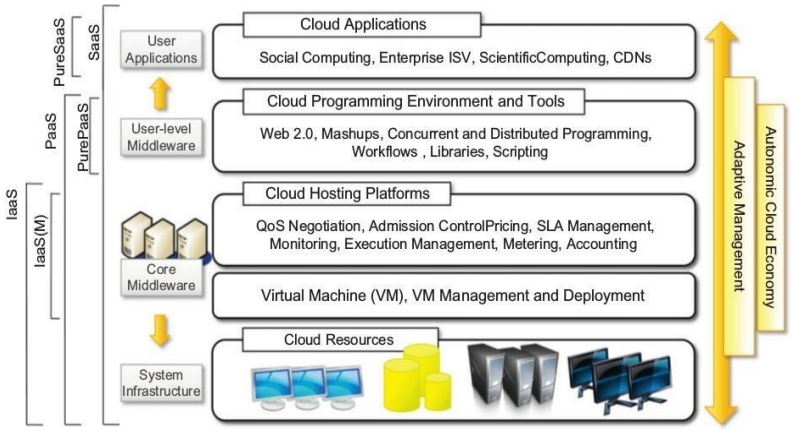


FIGURE 4.1

The cloud computing architecture.

Q7 b) Explain with figure Iaas reference implementation along with figure

Infrastructure- and Hardware-as-a-Service (IaaS/HaaS) solutions are the most popular and developed market segment of cloud computing. They deliver customizable infrastructure on demand. The available options within the IaaS offering umbrella range from single servers to entire infrastructures, including network devices, load balancers, and database and Web servers. The main technology used to deliver and implement these solutions is hardware virtualization: one or more virtual machines opportunely configured and interconnected define the distributed system on top of which applications are installed and deployed. Virtual machines also constitute the atomic components that are deployed and priced according to the specific features of the virtual hardware: memory, number of processors, and disk

storage. IaaS/HaaS solutions bring all the benefits of hardware virtualization: workload partitioning, application isolation, sandboxing, and hardware tuning.

From the perspective of the service provider, IaaS/HaaS allows better exploiting the IT infrastructure and provides a more secure environment where executing third party applications. From the perspective of the customer it reduces the administration and maintenance cost as well as the capital costs allocated to purchase hardware. At the same time, users can take advantage of the full customization offered by virtualization to deploy their infrastructure in the cloud; in most cases virtual machines come with only the selected operating system installed and the system can be configured with all the required packages and applications. Other solutions provide prepackaged system images that already contain the software stack required for the most common uses: Web servers, database servers, or LAMP1 stacks. Besides the basic virtual machine management capabilities, additional services can be provided, generally including the following: SLA resource-based allocation, workload management, support for infrastructure design through advanced Web interfaces, and the ability to integrate third-party IaaS solutions. Figure 4.2 provides an overall view of the components forming an Infrastructure-as-a-Service solution. It is possible to distinguish three principal layers: the physical infrastructure, the software management infrastructure, and the user interface. At the top layer the user interface provides access to the services exposed by the software management infrastructure. Such an interface is generally based on Web 2.0 technologies: Web services, RESTful APIs, and mash-ups.

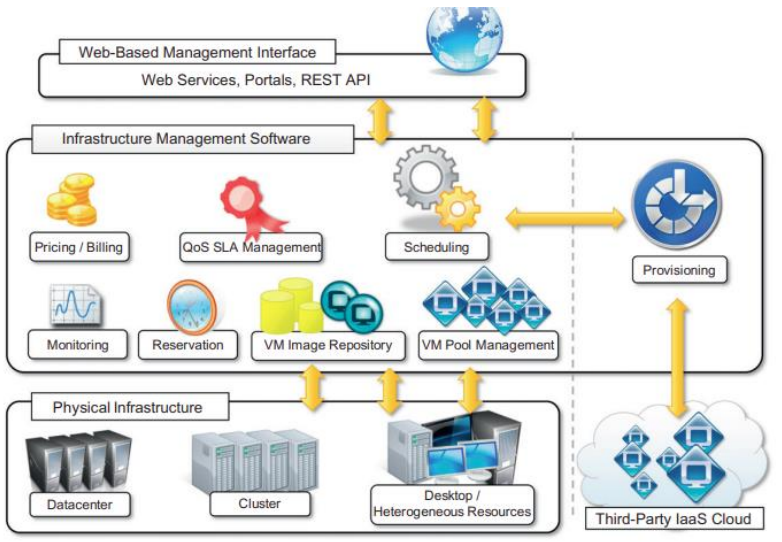


FIGURE 4.2
Infrastructure-as-a-Service reference implementation.

Q8a) Classify and explain various types of cloud

A more useful classification is given according to the administrative domain of a cloud. It is then possible to differentiate four different types of cloud:

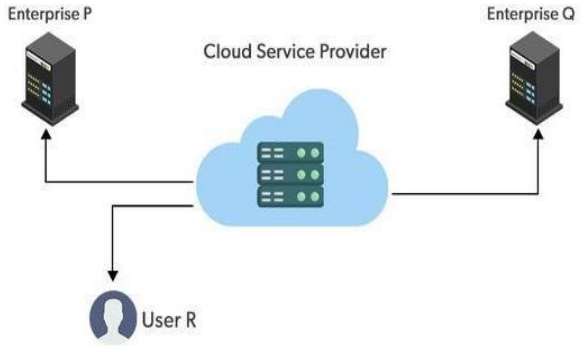
1. Public clouds. The cloud is open to the wider public.
2. Private clouds. The cloud is implemented within the private premises of an institution and generally made accessible to the members of the institution or a subset of them.
3. Hybrid or Heterogeneous clouds. The cloud is a combination of the two previous (Public + Private) solutions and most likely identifies a private cloud that has been augmented with resources or services hosted in a public cloud.
4. Community clouds. The cloud is characterized by a multi-administrative domain involving different deployment models (public, private, and hybrid), and it is specifically designed to address the needs of a specific industry.

Public Cloud

Public clouds are a realization of the canonical view of cloud computing in which the services offered are made available to anyone, from anywhere, and at any time through the Internet.

Public clouds are managed by third parties which provide cloud services over the internet to the public, these services are available as pay-as-you-go billing models.

Public clouds offer solutions for minimizing IT infrastructure costs and become a good optionfor handling peak loads on the local infrastructure. Public clouds are the go-to option for small enterprises, which are able to start their businesses without large upfront investments by completely relying on public infrastructure for their IT needs.



Private Cloud

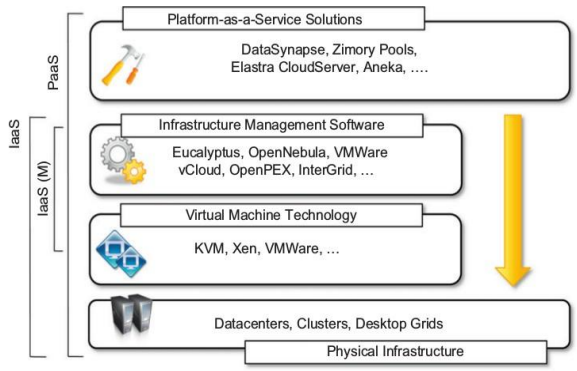


FIGURE 4.4
Private clouds hardware and software stack.

Figure 4.4 provides a comprehensive view of the solutions together with some reference to the most popular software used to deploy private clouds.

At the bottom layer of the software stack, virtual machine technologies such as Xen, KVM, and VMware serve as the foundations of the cloud.

Virtual machine management technologies such as VMware vCloud, Eucalyptus, and Open Nebula can be used to control the virtual infrastructure.

Private clouds are virtual distributed systems that rely on a private infrastructure and provide internal users with dynamic provisioning of computing resources. Instead of a pay-as-you-go model as in public clouds, there could be other schemes in place, considering the usage of the cloud and proportionally billing the different departments or sections of an enterprise.

Hybrid Cloud

A hybrid cloud is a heterogeneous distributed system formed by combining facilities of public cloud and private cloud.

A major drawback of private deployments is the inability to scale on-demand and efficiently address peak loads. Here public clouds are needed. Hence, a hybrid cloud takes advantage of both public and private clouds.

Figure 4.5 provides a general overview of a hybrid cloud: It is a heterogeneous distributed system resulting from a private cloud that integrates additional services or resources from one or more public and private clouds. For this reason, they are also called heterogeneous clouds.

As depicted in the diagram, dynamic provisioning is a fundamental component in this scenario. Hybrid clouds address scalability issues by leveraging external resources for exceeding capacitydemand. These resources or services are temporarily leased for the time required and then

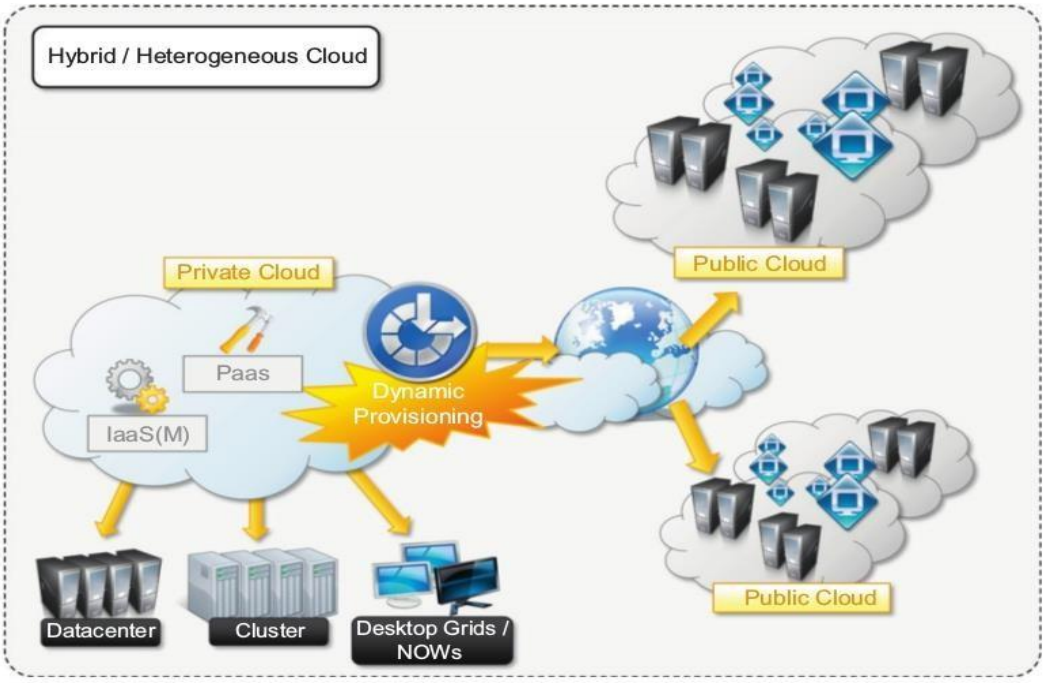


FIGURE 4.5

Hybrid/heterogeneous cloud overview.

released. This practice is also known as cloud-bursting.

Community clouds

Community clouds are distributed systems created by integrating the services of different clouds to address the specific needs of an industry, a community, or a business sector.

The National Institute of Standards and Technologies (NIST) characterizes community clouds as follows:

> *"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 premise or off premise."*

Or

> *In the community cloud, the infrastructure is shared between organizations that have shared concerns or tasks. The cloud may be managed by an organization or a third party.*

Figure4.6 provides a general view of the usage scenario of community clouds, together with reference architecture. The users of a specific community cloud fall in to a well-identified community, sharing the

same concerns or needs; they can be government bodies, industries, oreven simple users, but all of them focus on the same issues for their interaction with the cloud.

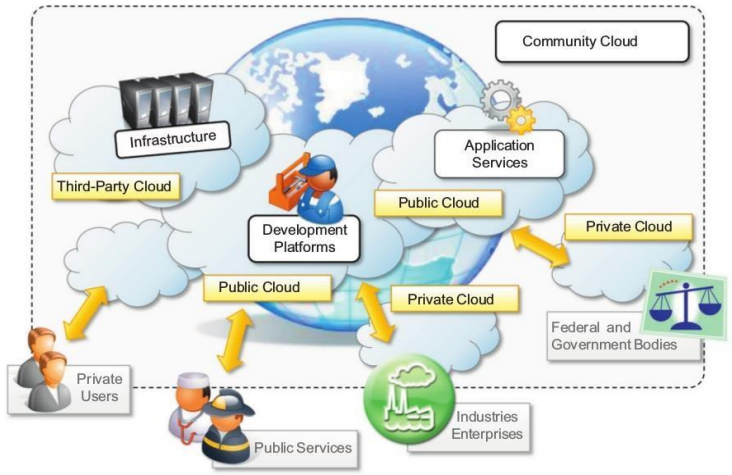


FIGURE 4.6
A community cloud.

Q8 b) Discuss the open challenges in cloud computing?

Cloud definition

As discussed earlier, there have been several attempts made to define cloud computing and to provide a classification of all the services and technologies identified as such. One of the most comprehensive formalizations is noted in the NIST working definition of cloud computing. It characterizes cloud computing as on-demand self-service, broad network access, resource-pooling, rapid elasticity, and measured service; classifies services as SaaS, PaaS, and IaaS; and categorizes deployment models as public, private, community, and hybrid clouds. The view is in line with our discussion and shared by many IT practitioners and academics.

Cloud interoperability and standards

Cloud computing is a service-based model for delivering IT infrastructure and applications like utilities such as power, water, and electricity. To fully realize this goal, introducing standards and allowing interoperability between solutions offered by different vendors are objectives of fundamental importance. Vendor lock-in constitutes one of the major strategic barriers against the seamless adoption of cloud computing at all stages. In particular there is major fear on the part of enterprises in which IT constitutes the significant part of their revenues. Vendor lock-in can prevent a customer from switching to another competitor's solution, or when this is possible, it happens at considerable conversion cost and requires significant amounts of time. This can occur either because the customer wants to find a more suitable solution for customer needs or because the vendor is no longer able to provide the required service. The presence of standards that are actually implemented and adopted in the cloud

computing community could give room for interoperability and then lessen the risks resulting from vendor lock-in.

Scalability and fault tolerance

The ability to scale on demand constitutes one of the most attractive features of cloud computing. Clouds allow scaling beyond the limits of the existing in-house IT resources, whether they are infrastructure (compute and storage) or applications services. To implement such a capability, the cloud middleware has to be designed with the principle of scalability along different dimensions in mind—for example, performance, size, and load. The cloud middleware manages a huge number of resource and users, which rely on the cloud to obtain the horsepower that they cannot obtain within the premises without bearing considerable administrative and maintenance costs. These costs are a reality for whomever develops, manages, and maintains the cloud middleware and offers the service to customers. In this scenario, the ability to tolerate failure becomes fundamental, sometimes even more important than providing an extremely efficient and optimized system. Hence, the challenge in this case is designing highly scalable and fault-tolerant systems that are easy to manage and at the same time provide competitive performance.

Security, trust, and privacy

Security, trust, and privacy issues are major obstacles for massive adoption of cloud computing. The traditional cryptographic technologies are used to prevent data tampering and access to sensitive information. The massive use of virtualization technologies exposes the existing system to new threats, which previously were not considered applicable. For example, it might be possible that applications hosted in the cloud can process sensitive information; such information can be stored within a cloud storage facility using the most advanced technology in cryptography to protect data and then be considered safe from any attempt to access it without the required permissions. Although these data are processed in memory, they must necessarily be decrypted by the legitimate application, but since the application is hosted in a managed virtual environment it becomes accessible to the virtual machine manager that by program is designed to access the memory pages of such an application. In this case, what is experienced is a lack of control over the environment in which the application is executed, which is made possible by leveraging the cloud. It then happens that a new way of using existing technologies creates new opportunities for additional threats to the security of applications. The lack of control over their own data and processes also poses severe problems for the trust we give to the cloud service provider and the level of privacy we want to have for our data.

Q9a) List and explain compute services of Amazon web services.

EC2 instances represent virtual machines. They are created using AMI as templates, which are specialized by selecting the number of cores, their computing power, and the installed memory

- The processing power is expressed in terms of virtual cores and EC2 Compute Units (ECUs). The ECU is a measure of the computing power of a virtual core; it is used to express a predictable quantity of real CPU power that is allocated to an instance.
- By using compute units instead of real frequency values, Amazon can change over time the mapping of such units to the underlying real amount of computing power allocated, thus keeping the performance of EC2 instances consistent with standards set by the times.
- Over time, the hardware supporting the underlying infrastructure will be replaced by more powerful hardware, and the use of ECUs helps give users a consistent view of the performance offered by EC2 instances.
- Since users rent computing capacity rather than buying hardware, this approach is reasonable
- Six major currently available configurations categories for EC2 instances.:
  - Standard instances. This class offers a set of configurations that are suitable for most applications. EC2 provides three different categories of increasing computing power, storage, and memory.
  - Micro instances. This class is suitable for those applications that consume a limited amount of computing power and memory and occasionally need bursts in CPU cycles to process surges in the workload. Micro instances can be used for small Web applications with limited traffic.
  - High-memory instances. This class targets applications that need to process huge workloads and require large amounts of memory.
  - High-CPU instances. This class targets compute-intensive applications. Two configurations are available where computing power proportionally increases more than memory.
  - Cluster Compute instances. This class is used to provide virtual cluster services. Instances in this category are characterized by high CPU compute power and large memory and an extremely high I/O and network performance, which makes it suitable for HPC applications.
  - Cluster GPU instances. This class provides instances featuring graphic processing units (GPUs) and high compute power, large memory, and extremely high I/O and network performance..

- EC2 instances are priced hourly according to the category they belong to. At the beginning of every hour of usage, the user will be charged the cost of the entire hour. The hourly expense charged for one instance is constant.
- Another alternative is represented by spot instances. These instances are much more dynamic in terms of pricing and lifetime since they are made available to the user according to the load of EC2 and the availability of resources.
- EC2 instances can be run either by using the command-line tools provided by Amazon, which connects the Amazon Web Service that provides remote access to the EC2 infrastructure, or via the AWS console, which allows the management of other services, such as S3

Q9b) Explain Google AppEngine with platform architecture

Google AppEngine is a PaaS implementation that provides services for developing and hosting scalable Web applications. AppEngine is essentially a distributed and scalable runtime environment that leverages Google's distributed infrastructure to scale out applications facing a large number of requests by allocating more computing resources to them and balancing the load among them.

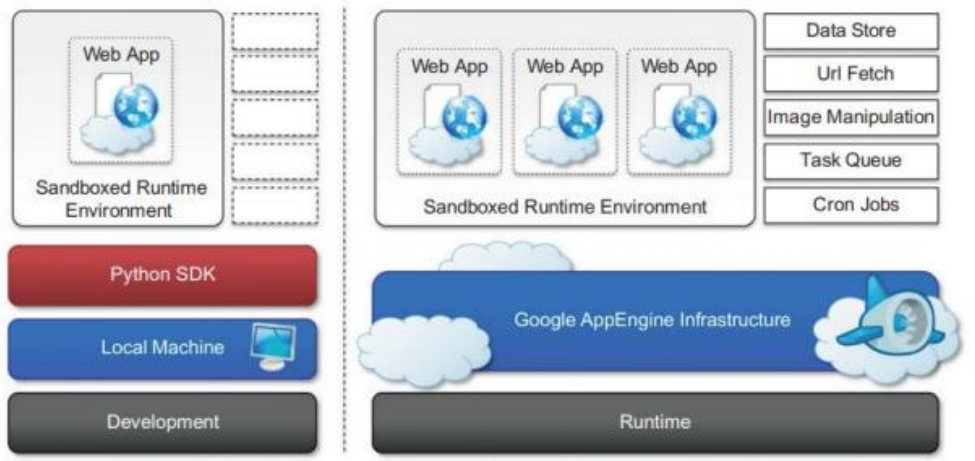


FIGURE 9.2

Google AppEngine platform architecture.

1) Infrastructure

AppEngine hosts Web applications, and its primary function is to serve users requests efficiently. To do so, AppEngine's infrastructure takes advantage of many servers available within Google datacenters.

2) Runtime environment

- The runtime environment represents the execution context of applications hosted on AppEngine. With reference to the AppEngine infrastructure code, which is always active and running, the runtime comes into existence when the request handler starts executing and terminates once the handler has completed.
- Sandboxing One of the major responsibilities of the runtime environment is to provide the application environment with an isolated and protected context in which it can execute without causing a threat to the server and without being influenced by other applications.
- Supported runtimes: Currently, it is possible to develop AppEngine applications using three different languages and related technologies: Java, Python, and Go. AppEngine currently supports Java 6, and developers can use the common tools for Web application development in Java, such as the Java Server Pages (JSP), and the applications interact with the environment by using the Java Servlet standard

3) Storage

AppEngine provides various types of storage, which operate differently depending on the volatility of the data. There are three different levels of storage: in memory-cache, storage for semistructured data, and long-term storage for static data.

- Static file servers : Web applications are composed of dynamic and static data. Dynamic data are a result of the logic of the application and the interaction with the user.
- DataStore is a service that allows developers to store semistructured data. The service is designed to scale and optimized to quickly access data. DataStore can be considered as a large object database in which to store objects that can be retrieved by a specified key. Both the type of the key and the structure of the object can vary.

4) Application services

Applications hosted on AppEngine take the most from the services made available through the runtime environment. These services simplify most of the common operations that are performed in Web

applications: access to data, account management, integration of external resources, messaging and communication, image manipulation, and asynchronous computation.

- UrlFetch : The sandbox environment does not allow applications to open arbitrary connections through sockets, but it does provide developers with the capability of retrieving a remote resource through HTTP/HTTPS by means of the UrlFetch service.
- MemCache : AppEngine provides caching services by means of MemCache. This is a distributed in-memory cache that is optimized for fast access and provides developers with a volatile store for the objects that are frequently accessed.
- Mail and instant messaging: AppEngine provides developers with the ability to send and receive mails through Mail. The service allows sending email on behalf of the application to specific user accounts.
- Account management: AppEngine simplifies account management by allowing developers to leverage Google account management by means of Google Accounts.
- Image manipulation: AppEngine allows applications to perform image resizing, rotation, mirroring, and enhancement by means of Image Manipulation, a service that is also used in other Google products.

5) Compute services

AppEngine offers additional services such as Task Queues and Cron Jobs that simplify the execution of computations that are off-bandwidth or those that cannot be performed within the timeframe of the Web request.

- Task queues allow applications to submit a task for a later execution.
- Cron jobs : it is possible to schedule the required operation at the desired time by using the Cron Jobs service.

Q10a) Illustrate with figure cloud environment for satellite data processing

Geoscience applications collect, produce, and analyze massive amounts of geospatial and nonspatial data. As the technology progresses and our planet becomes more instrumented, volume of data that needs to be processed increases significantly.

Geographic information system (GIS) applications capture, store, manipulate, analyze, manage, and present all types of geographically referenced data.

Cloud computing is an attractive option for executing these demanding tasks and extracting meaningful information to support decision makers.

Satellite remote sensing generates hundreds of gigabytes of raw images that need to be further processed to become the basis of several different GIS products. This process requires both I/O and compute-intensive tasks. Large images need to be moved from a ground station's local storage to compute facilities, where several transformations and corrections are applied.

The system shown in Figure 10.4 integrates several technologies across the entire computing stack.

A SaaS application provides a collection of services for such tasks as geocode generation and

data visualization.

At the PaaS level, Aneka controls the importing of data into the virtualized infrastructure and the execution of image-processing tasks that produce the desired outcome from raw satellite images. The platform leverages a Xen private cloud and the Aneka technology to dynamically provisionthe required resources.

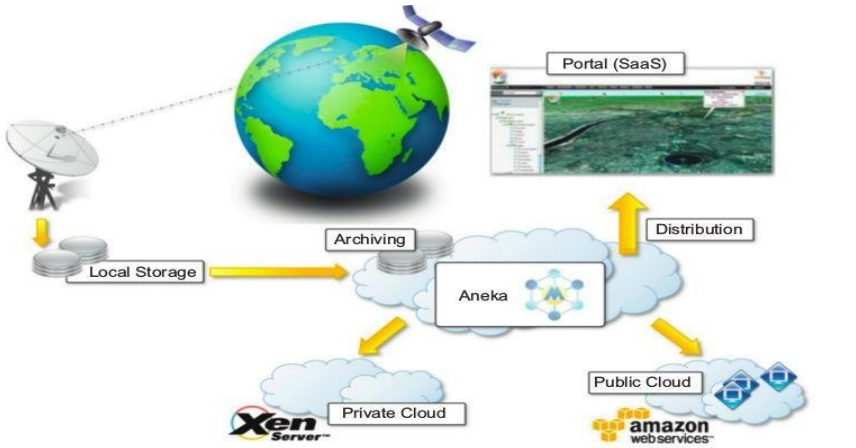


FIGURE 10.4
A cloud environment for satellite data processing.

Q10b) Explain DropBox and iCloud application with figure

Dropbox and Icloud

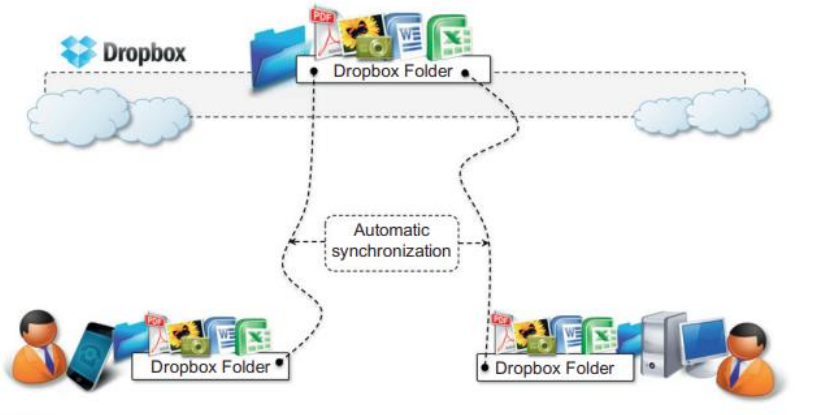


FIGURE 10.6
Dropbox usage scenario.

The most popular solution for online document storage is Dropbox, an online application that allows users to synchronize any file across any platform and any device in a seamless manner (see Figure 10.6).

Dropbox provides users with a free amount of storage that is accessible through the abstraction of a folder. Users can either access their Dropbox folder through a browser or by downloading and installing a Dropbox client, which provides access to the online storage by means of a special folder. All the modifications into this folder are silently synched so that changes are notified to all the local instances of the Dropbox folder across all the devices.