CMR
INSTITUTE OF
TECHNOLOGY

USN □□□□□□□□□□

Internal Assessment Test 2– Feb. 2023

Sub:	**Cloud Computing**							**Sub Code:**	**20MCA342**
Date:	**7/2/2023**	**Duration:**	**90 min's**	**Max Marks:**	**50**	**Sem:**	**III**	**Branch:**	**MCA**

Note : Answer FIVE FULL Questions, choosing ONE full question from each Module

		MARKS	OBE	
			CO	RBT
	PART I			
1	What is virtualization and what are its benefits?	[10]	CO5	L2
	OR			
2	Discuss in detail about characteristics of virtualized solutions.	[10]	CO5	L2
	PART II			
3	Discuss the machine reference model of execution virtualization	[10]	CO5	L3
	OR			
4	List and discuss different types of hardware virtualization techniques	[10]	CO5	L3
	PART III			
5	What are the problems of virtualization and how to solve them?	[10]	CO5	L4
	OR			
6	Discuss and clarify in detail about need for a VMware full virtualization reference model	[10]	CO5	L3
	PART IV			
7	Explain with suitable diagram the cloud computing architecture.	[10]	CO3	L3
	OR			
8	Explain platform as a service reference model.	[10]	CO3	L3
	PART V			
9	With a neat diagram explain Private cloud	[10]	CO3	L2
	OR			
10	With a neat diagram explain Public cloud.	[10]	CO3	L2

Q1) What is virtualization and what are its benefits?

Virtualization allows the creation of a secure, customizable, and isolated execution environment for running applications, even if they are untrusted, without affecting other users' applications. The basis of this technology is the ability of a computer program—or a combination of software and hardware—to emulate an executing environment separate from the one that hosts such programs.

Virtualization is a large umbrella of technologies and concepts that are meant to provide an abstract environment—whether virtual hardware or an operating system—to run applications.

Hardware virtualization plays a fundamental role in efficiently delivering Infrastructure-as-a-Service (IaaS) solutions for cloud computing

Virtualization technology provide a virtual environment for not only executing applications but also for storage, memory, and networking

- **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

Q2) Discuss in detail about characteristics of virtualized solutions

1) Increased security

- The ability to control the execution of a guest in a completely transparent manner opens new possibilities for delivering a secure, controlled execution environment.
- The virtual machine represents an emulated environment in which the guest is executed.
- All the operations of the guest are generally performed against the virtual machine
- Resources exposed by the host can then be hidden or simply protected from the guest
- sensitive information that is contained in the host can be naturally hidden without the need to install complex security policies.
- Increased security is a requirement when dealing with untrusted code (Example: applets downloaded from the Internet run in a sandboxed version of JVM)

2) Managed execution

- Virtualization of the execution environment not only allows increased security, but a wider range of features also can be implemented like sharing, aggregation, emulation, and isolation are the most relevant features
  - Sharing: a particularly important feature in virtualized data centers, used to reduce the number of active servers and limit power consumption.
  - Aggregation: A group of separate hosts can be tied together and represented to guests as a single virtual host

- Emulation: a completely different environment with respect to the host can be emulated, thus allowing the execution of guest programs requiring specific characteristics that are not present in the physical host
- Isolation: Virtualization allows providing guests with a completely separate environment, in which they are executed. The guest program performs its activity by interacting with an abstraction layer, which provides access to the underlying resources

  Advantages of Isolation:

  - it provides a separation between the host and the guest
  - it allows multiple guests to run on the same host without interfering with each other

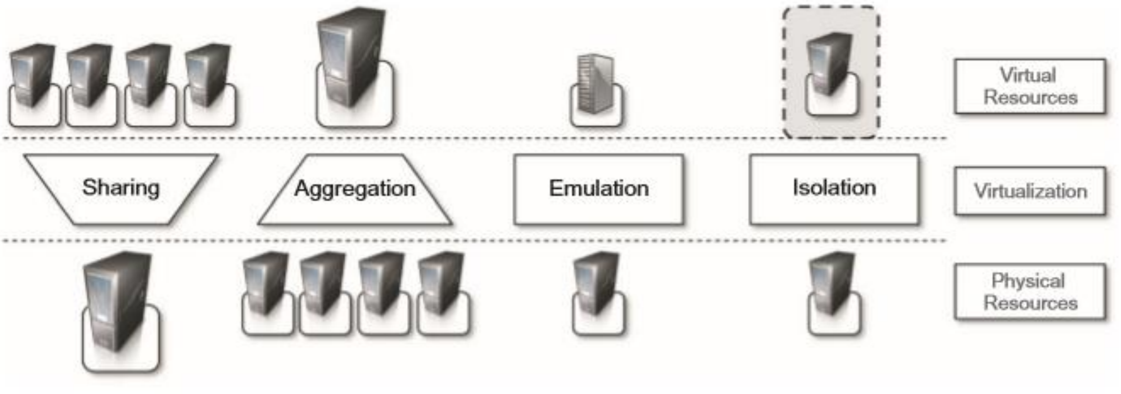


FIGURE 3.2

Functions enabled by managed execution.

3) Portability

- Portability allows having your own system always with you and ready to use as long as the required virtual machine manager is available
- the guest is packaged into a virtual image that, in most cases, can be safely moved and executed on top of different virtual machines
- Virtual images are generally proprietary formats that require a specific virtual machine manager to be executed
- In the case of programming-level virtualization, the binary code representing application components (jars or assemblies) can be run without any recompilation on any implementation of the corresponding virtual machine

Q3) Discuss the machine reference model of execution virtualization

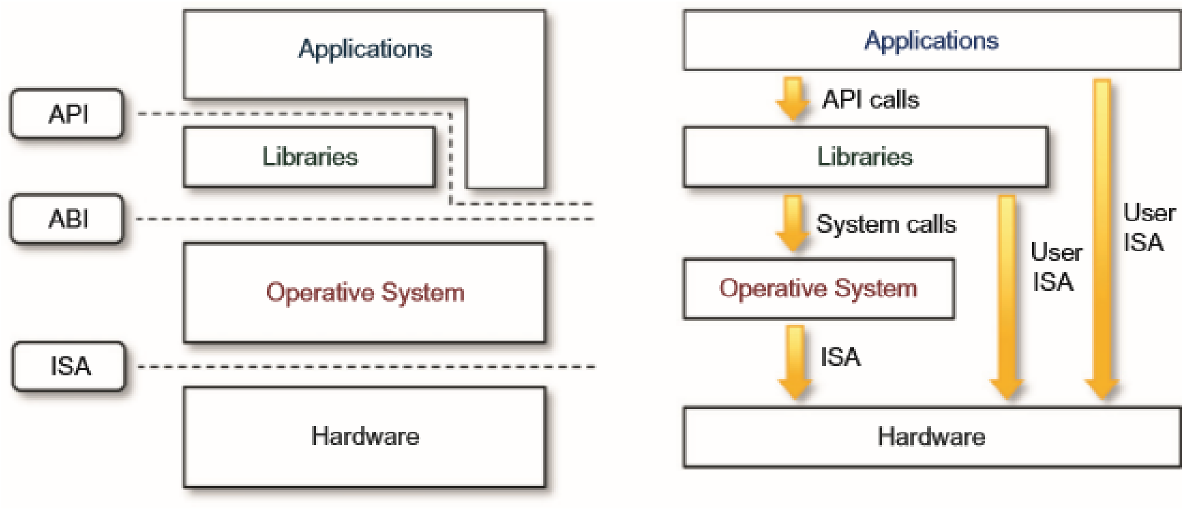


FIGURE 3.4

A machine reference model.

Virtualizing an execution environment at different levels of the computing stack requires a reference model that defines the interfaces between the levels of abstractions, which hide implementation details. From this perspective, virtualization techniques actually replace one of the layers and intercept the calls that are directed toward it. Therefore, a clear separation between layers simplifies their implementation, which only requires the emulation of the interfaces and a proper interaction with the underlying layer. Modern computing systems can be expressed in terms of the reference model described in Figure 3.4. At the bottom layer, the model for the hardware is expressed in terms of the Instruction Set Architecture (ISA), which defines the instruction set for the processor, registers, memory, and interrupt management. ISA is the interface between hardware and software, and it is important to the operating system (OS) developer (System ISA) and developers of applications that directly manage the underlying hardware (User ISA). The application binary interface (ABI) separates the operating system layer from the applications and libraries, which are managed by the OS. ABI covers details such as low-level data types, alignment, and call conventions and defines a format for executable programs. System calls are defined at this level. This interface allows portability of applications and libraries across operating systems that implement the same ABI. The highest level of abstraction is represented by the application programming interface (API), which interfaces applications to libraries and/or the underlying operating system. For any operation to be performed in the application level API, ABI and ISA are responsible for making it happen. The high-level abstraction is converted into machine-level instructions to perform the actual operations supported by the processor. The machine-level resources, such as processor registers and main memory capacities, are used to perform the operation at the hardware level of the central processing unit (CPU). This layered approach simplifies the development and implementation of computing systems and simplifies the implementation of multitasking and the coexistence of multiple executing environments. In fact, such a model not only requires limited knowledge of the entire computing stack, but it also provides ways to implement a minimal security model for managing and accessing shared resources.

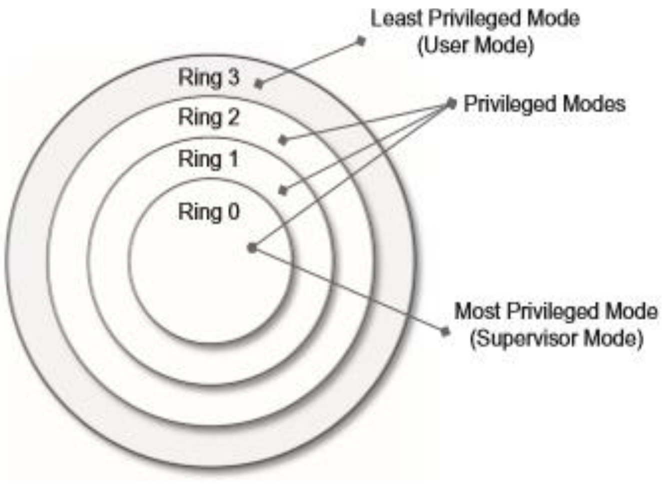


FIGURE 3.5

Security rings and privilege modes.

Hardware has been divided into different security classes that define who can operate with them.

The first distinction can be made between privileged and non privileged instructions.

Non privileged instructions are those instructions that can be used without interfering with other tasks because they do not access shared resources.

Privileged instructions are those that are executed under specific restrictions and are mostly used for sensitive operations, which expose (behavior-sensitive) or modify (control-sensitive) the privileged state.

For instance, a possible implementation features a hierarchy of privileges (see Figure 3.5) in the form of ring-based security: Ring 0, Ring 1, Ring 2, and Ring 3; Ring 0 is in the most privileged level and Ring 3 in the least privileged level.

Ring 0 is used by the kernel of the OS, rings 1 and 2 are used by the OS-level services, and Ring 3 is used by the user.

Recent systems support only two levels, with Ring 0 for supervisor mode and Ring 3 for user mode.

Q4) List and discuss different types of hardware virtualization techniques

1) Hardware-assisted virtualization.

This term refers to a scenario in which the hardware provides architectural support for building a virtual machine manager able to run a guest operating system in complete isolation. This technique was originally introduced in the IBM System/370. At present, examples of hardware-assisted virtualization are the extensions to the x86-64 bit architecture introduced with Intel VT (formerly known as Vanderpool) and

AMD V (formerly known as Pacifica). These extensions, which differ between the two vendors, are meant to reduce the performance penalties experienced by emulating x86 hardware with hypervisors.

Products such as VMware Virtual Platform, introduced in 1999 by VMware, which pioneered the field of x86 virtualization, were based on this technique. After 2006, Intel and AMD introduced processor extensions, and a wide range of virtualization solutions took advantage of them: Kernel-based Virtual Machine (KVM), VirtualBox, Xen, VMware, Hyper-V, Sun xVM, Parallels, and others.

2) Full virtualization.

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 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.

3) Paravirtualization.

This is a not-transparent virtualization solution that allows implementing thin virtual machine managers. Paravirtualization techniques expose a software interface to the virtual machine that is slightly modified from the host and, as a consequence, guests need to be modified. The aim of paravirtualization is to provide the capability to demand the execution of performance-critical operations directly on the host, thus preventing performance losses that would otherwise be experienced in managed execution. This allows a simpler implementation of virtual machine managers that have to simply transfer the execution of these operations, which were hard to virtualize, directly to the host. To take advantage of such an opportunity, guest operating systems need to be modified and explicitly ported by remapping the performance-critical operations through the virtual machine software interface. This is possible when the source code of the operating system is available, and this is the reason that paravirtualization was mostly explored in the opensource and academic environment. This technique has been successfully used by Xen for providing virtualization solutions for Linux-based operating systems specifically ported to run on Xen hypervisors.

Other solutions using paravirtualization include VMWare, Parallels, and some solutions for embedded and real-time environments such as TRANGO, Wind River, and XtratuM

4) Partial virtualization.

Partial virtualization provides a partial emulation of the underlying hardware, thus not allowing the complete execution of the guest operating system in complete isolation. Partial virtualization allows many applications to run transparently, but not all the features of the operating system can be supported, as happens with full virtualization. An example of partial virtualization is address space virtualization used in time-sharing systems; this allows multiple applications and users to run concurrently in a separate memory space, but they still share the same hardware resources (disk, processor, and network). Historically, partial virtualization has been an important milestone for achieving full virtualization, and it was implemented on the experimental IBM M44/44X. Address space virtualization is a common feature of contemporary operating systems

Q5) What are the problems of virtualization and how to solve them?

1) Performance degradation

Performance is definitely one of the major concerns in using virtualization technology. Since virtualization interposes an abstraction layer between the guest and the host, the guest can experience increased latencies. For instance, in the case of hardware virtualization, where the intermediate emulates a bare machine on top of which an entire system can be installed, the causes of performance degradation can be traced back to the overhead introduced by the following activities:

- Maintaining the status of virtual processors
- Support of privileged instructions (trap and simulate privileged instructions)
- Support of paging within VM
- Console functions

Furthermore, when hardware virtualization is realized through a program that is installed or executed on top of the host operating systems, a major source of performance degradation is represented by the fact that the virtual machine manager is executed and scheduled together with other applications, thus sharing with them the resources of the host.

Similar consideration can be made in the case of virtualization technologies at higher levels, such as in the case of programming language virtual machines (Java, .NET, and others). Binary translation and interpretation can slow down the execution of managed applications. Moreover, because their execution is filtered by the runtime environment, access to memory and other physical resources can represent sources of performance degradation.

These concerns are becoming less and less important thanks to technology advancements and the ever-increasing computational power available today. For example, specific techniques for hardware virtualization such as paravirtualization can increase the performance of the guest program by offloading most of its execution to the host without any change. In programming-level virtual machines such as the JVM or .NET, compilation to native code is offered as an option when performance is a serious concern

2) Inefficiency and degraded user experience

Virtualization can sometime lead to an inefficient use of the host. In particular, some of the specific features of the host cannot be exposed by the abstraction layer and then become inaccessible. In the case of hardware virtualization, this could happen for device drivers: The virtual machine can sometime simply provide a default graphic card that maps only a subset of the features available in the host. In the case of programming-level virtual machines, some of the features of the underlying operating systems may become inaccessible unless specific libraries are used. For example, in the first version of Java the support for graphic programming was very limited and the look and feel of applications was very poor compared to native applications. These issues have been resolved by providing a new framework called Swing for designing the user interface, and further improvements have been done by integrating support for the OpenGL libraries in the software development kit.

3) Security holes and new threats

Virtualization opens the door to a new and unexpected form of phishing. The capability of emulating a host in a completely transparent manner led the way to malicious programs that are designed to extract sensitive information from the guest. In the case of hardware virtualization, malicious programs can preload themselves before the operating system and act as a thin virtual machine manager toward it. The operating system is then controlled and can be manipulated to extract sensitive information of interest to third parties. The same considerations can be made for programming-level virtual machines: Modified versions of the runtime environment can access sensitive information or monitor the memory locations utilized by guest applications while these are executed. To make this possible, the original version of the runtime environment needs to be replaced by the modified one, which can generally happen if the malware is run within an administrative context or a security hole of the host operating system is exploited

Q6) Discuss and clarify in detail about need for a VMware full virtualization reference model

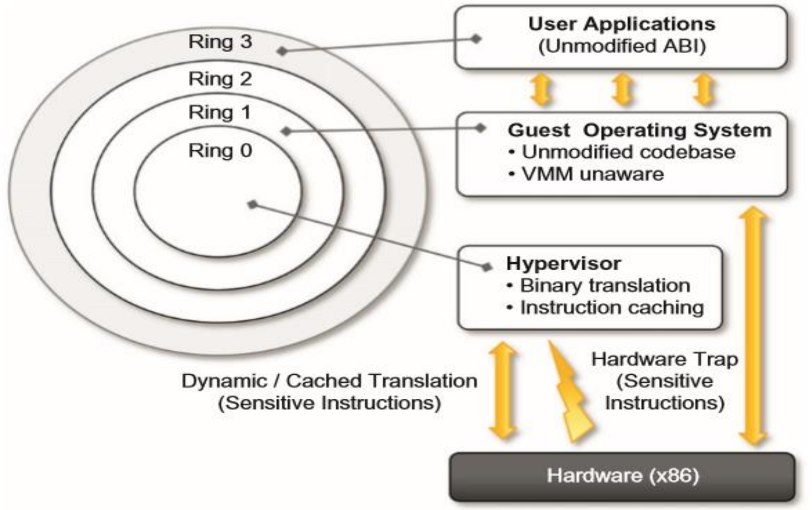


FIGURE 3.12

A full virtualization reference model.

VMware is well known for the capability to virtualize x86 architectures, which runs unmodified on top of their hypervisors. With the new generation of hardware architectures and the introduction of hardware-assisted virtualization (Intel VT-x and AMD V) in 2006, full virtualization is made possible with hardware support, but before that date, the use of dynamic binary translation was the only solution that allowed running x86 guest operating systems unmodified in a virtualized environment

As discussed before, x86 architecture design does not satisfy the first theorem of virtualization, since the set of sensitive instructions is not a subset of the privileged instructions. This causes a different behavior when such instructions are not executed in Ring 0, which is the normal case in a virtualization scenario where the guest OS is run in Ring 1. Generally, a trap is generated and the way it is managed differentiates the solutions in which virtualization is implemented for x86 hardware. In the case of dynamic binary translation, the trap triggers the translation of the offending instructions into an equivalent set of instructions that achieves the same goal without generating exceptions. Moreover, to improve performance, the equivalent set of instruction is cached so that translation is no longer necessary for further occurrences of the same instructions.

The major advantage is that guests can run unmodified in a virtualized environment, which is a crucial feature for operating systems for which source code is not available

VMware achieves full virtualization by providing virtual representation of memory and I/O devices. Memory virtualization constitutes another challenge of virtualized environments and can deeply impact performance without the appropriate hardware support. The main reason is the presence of a memory management unit (MMU), which needs to be emulated as part of the virtual hardware. Especially in the case of hosted hypervisors (Type II), where the virtual MMU and the host-OS MMU are traversed sequentially before getting to the physical memory page, the impact on performance can be significant. To avoid nested translation, the translation look-aside buffer (TLB) in the virtual MMU directly maps physical pages, and the performance slowdown only occurs in case of a TLB miss.

Finally, VMware also provides full virtualization of I/O devices such as network controllers and other peripherals such as keyboard, mouse, disks, and universal serial bus (USB) controllers.

Q7) Explain with suitable diagram the cloud computing architecture

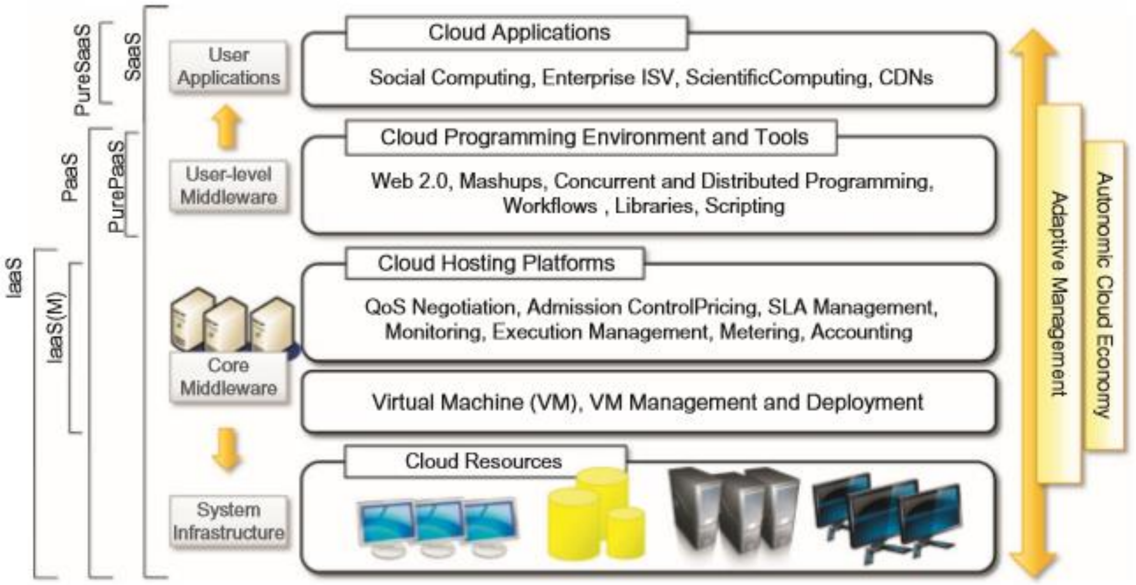

PureSaaS
SaaS
PaaS
PurePaaS
IaaS
IaaS(M)
User Applications
User-level Middleware
Core Middleware
System Infrastructure
Cloud Applications
Social Computing, Enterprise ISV, ScientificComputing, CDNs
Cloud Programming Environment and Tools
Web 2.0, Mashups, Concurrent and Distributed Programming, Workflows , Libraries, Scripting
Cloud Hosting Platforms
QoS Negotiation, Admission ControlPricing, SLA Management, Monitoring, Execution Management, Metering, Accounting
Virtual Machine (VM), VM Management and Deployment
Cloud Resources
Adaptive Management
Autonomic Cloud Economy

FIGURE 4.1

The cloud computing architecture.

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 commonly 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 to finely partition the hardware resources such as CPU and memory and to virtualize specific devices, thus meeting the requirements of users and applications. This solution is generally paired with storage and network virtualization strategies, which allow the infrastructure to be completely virtualized and controlled.

The combination of cloud hosting platforms and resources is generally classified as a 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)). In this second case, the management layer is often integrated with other IaaS solutions that provide physical infrastructure and adds value to them.

IaaS solutions are suitable for designing the system infrastructure but provide limited services to build applications. Such service is provided by cloud programming environments and tools, which form a new layer for offering users a development platform for applications. 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. PaaS solutions generally include the infrastructure as well, which is bundled as part of the service provided to users. In the case of Pure PaaS, only the user-level middleware is offered, and it has to be complemented with a virtual or physical infrastructure

The top layer of the reference model contains services delivered at the application level. These are mostly referred to 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. Other applications belonging to this layer are those that strongly leverage the Internet for their core functionalities that rely on the cloud to sustain a larger number of users; this is the case of gaming portals and, in general, social networking websites. As a vision, any service offered in the cloud computing style should be able to adaptively change and expose an autonomic behavior, in particular for its availability and performance. As a reference model, it is then expected to have an adaptive management layer in charge of elastically scaling on demand. SaaS implementations should feature such behavior automatically, whereas PaaS and IaaS generally provide this functionality as a part of the API exposed to users.

The reference model also introduces the concept of everything as a Service (XaaS). This is one of the most important elements of cloud computing: Cloud services from different providers can be combined to provide a completely integrated solution covering all the computing stack of a system.

Q8) Explain platform as a service reference model

- Platform-as-a-service (PaaS) provides a development and deployment platform for running applications in the cloud.
- They constitute the middleware on top of which applications are built.
- Application management is the core functionality of the middleware.
- o PaaS implementations provide applications with a runtime environment and do not expose any service for managing the underlying infrastructure.

- They automate the process of deploying applications to the infrastructure, configuring application components, provisioning and configuring supporting technologies such as load balancers and databases, and managing system change based on policies set by the user.

- From a user point of view, the core middleware exposes interfaces that allow programming and deploying applications on the cloud.
- These can be in the form of a Web-based interface or in the form of programming APIs and libraries

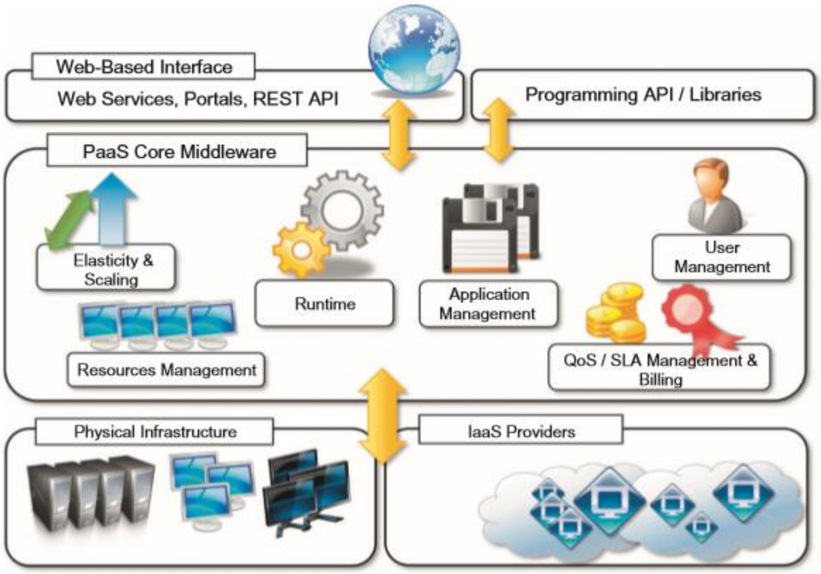


FIGURE 4.3

The Platform-as-a-Service reference model.

- Some implementations provide a completely Web-based interface hosted in the cloud and offering a variety of services
- PaaS solutions can offer middleware for developing applications together with the infrastructure (middleware + infrastructure[1]) or simply provide users with the software that is installed on the user premises (middleware)
  - 1 - the PaaS provider also owns large data-centers where applications are executed
  - 2 - the middleware constitutes the core value of the offering (PurePaaS)

Three wide categories of the most popular PaaS implementations

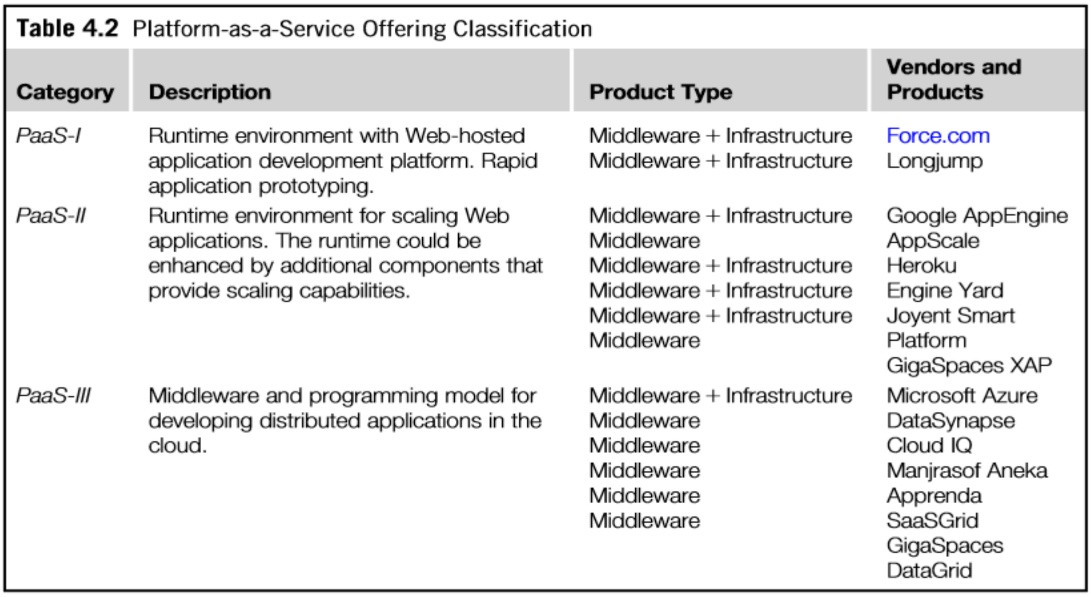


Table 4.2 Platform-as-a-Service Offering Classification

Category	Description	Product Type	Vendors and Products
PaaS-I	Runtime environment with Web-hosted application development platform. Rapid application prototyping.	Middleware + Infrastructure Middleware + Infrastructure	Force.com Longjump
PaaS-II	Runtime environment for scaling Web applications. The runtime could be enhanced by additional components that provide scaling capabilities.	Middleware + Infrastructure Middleware Middleware + Infrastructure Middleware + Infrastructure Middleware + Infrastructure Middleware	Google AppEngine AppScale Heroku Engine Yard Joyent Smart Platform GigaSpaces XAP
PaaS-III	Middleware and programming model for developing distributed applications in the cloud.	Middleware + Infrastructure Middleware Middleware Middleware Middleware Middleware	Microsoft Azure DataSynapse Cloud IQ Manjrasof Aneka Apprenda SaaSGrid GigaSpaces DataGrid

- PaaS-I:
  - completely follow the cloud computing style for application development and deployment
  - offer an Integrated Development Environment (IDE) hosted within the Web browser where applications are designed, developed, composed, and deployed
- PaaS-II:
  - Focused on providing a scalable infrastructure for Web application, mostly websites
  - Developers generally use the providers' APIs to develop applications
- PaaS-III: provide a cloud programming platform for any kind of application, not only Web applications

The PaaS umbrella encompasses a variety of solutions for developing and hosting applications in the cloud

Essential characteristics that identify a PaaS solution:

- Runtime framework: represents the "software stack" of the PaaS model; executes end-user code according to the policies set by the user and the provider
- Abstraction: PaaS solutions offer a way to deploy and manage applications on the cloud rather than a bunch of virtual machines on top of which the IT infrastructure is built and configured
- Automation: automate the process of deploying applications to the infrastructure, scaling them by provisioning additional resources when needed, according to the SLA made between the customers and the provider
- Cloud services: provide developers and architects with services and APIs helping them to simplify the creation and delivery of elastic and highly available cloud applications, which includes specific components for developing applications, advanced services for application monitoring, management, and reporting

Another essential component for a PaaS-based approach is the ability to integrate third-party cloud services offered from other vendors by leveraging service-oriented architecture, happen through interfaces and protocols.

PaaS environments deliver a platform for developing applications, which exposes a well-defined set of APIs and, in most cases, binds the application to the specific runtime of the PaaS provider

PaaS solutions can cut the cost across development, deployment, and management of applications

It helps management reduce the risk of ever-changing technologies by offloading the cost of upgrading the technology to the PaaS provider

The PaaS approach, when bundled with underlying IaaS solutions, helps even small start-up companies quickly offer customers integrated solutions on a hosted platform at a very minimal cost

Q9) With a neat diagram explain Private cloud

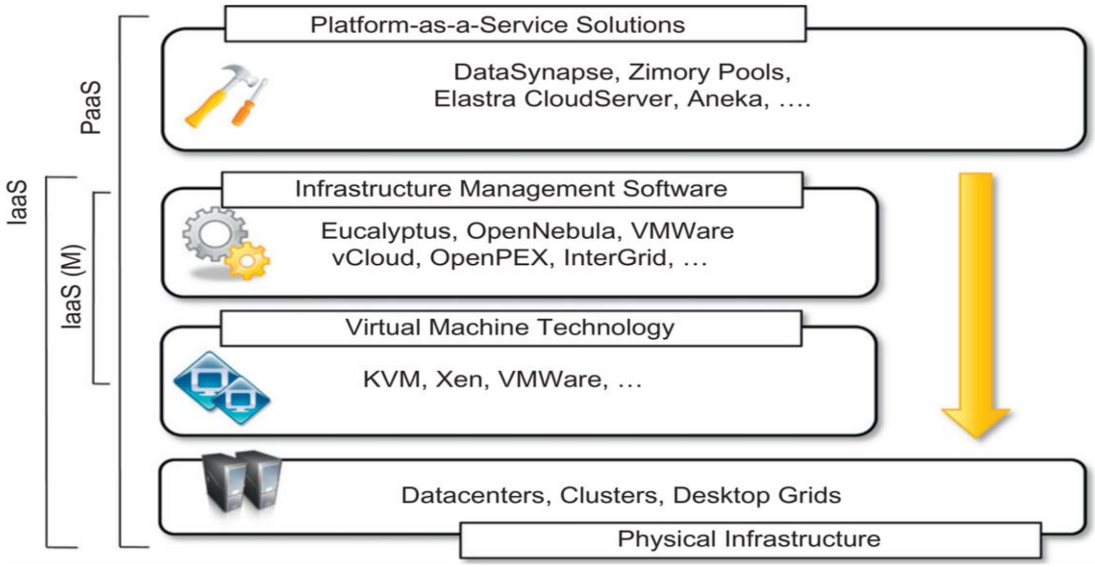
Private clouds

- 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, there could be other schemes in place, taking into account the usage of the cloud and proportionally billing the different departments or sections of an enterprise.
- Key Advantages:

○ Customer Information Protection: security concerns are less critical

○ Infrastructure ensuring SLAs: QoS by means of system monitoring and maintenance, and disaster recovery, and other uptime services

○ Compliance with standard procedures and operations: for third party specific procedures when deploying and executing applications


Platform-as-a-Service Solutions
DataSynapse, Zimory Pools,
Elastra CloudServer, Aneka, ….
Infrastructure Management Software
Eucalyptus, OpenNebula, VMWare
vCloud, OpenPEX, InterGrid, …
Virtual Machine Technology
KVM, Xen, VMWare, …
Datacenters, Clusters, Desktop Grids
Physical Infrastructure
PaaS
IaaS
IaaS (M)

Private clouds hardware and software stack

- datacenter, a cluster, an enterprise desktop grid: existing IT infrastructure already deployed on the private premises
- The physical layer is complemented with infrastructure management software or a PaaS solution, according to the service delivered to the users of the cloud
- bottom layer of the software stack: serve as the foundations of the cloud
- VMware vCloud, Eucalyptus, and OpenNebula: used to control the virtual infrastructure and provide an IaaS solution
- OpenNebula is an open-source solution for virtual infrastructure management
- Its modular architecture allows extending the software with additional features such as the capability of reserving virtual machine instances
- OpenPEX: Web-based system that allows the reservation of virtual machine instances and is designed to support different back ends
- InterGrid: provides added value on top of OpenNebula and Amazon EC2 by allowing the reservation of virtual machine instances and managing multi-administrative domain clouds
- PaaS solutions can provide an additional layer and deliver a high-level service for private clouds
- DataSynapse: global provider of application virtualization software; provides a flexible environment for building private clouds on top of datacenters
- Elastra Cloud Server: a platform for easily configuring and deploying distributed application infrastructures on clouds
- Zimory: provides a software infrastructure layer that automates the use of resource pools based on Xen, KVM, and VMware virtualization technologies.
- Aneka: a software development platform that can be used to deploy a cloud infrastructure on top of heterogeneous hardware: data-centers, clusters, and desktop grids.

- provides a pluggable service-oriented architecture that's mainly devoted to supporting the execution of distributed applications with different programming models: bag of tasks, MapReduce, and others.
- Private clouds can provide in-house solutions for cloud computing, but if compared to public clouds they exhibit more limited capability to scale elastically on demand

Q10) With a neat diagram explain Public cloud

Public clouds

- services offered are made available to anyone, from anywhere, and at any time through the Internet
- they are a distributed system, most likely composed of one or more data-centers connected together, on top of which the specific services offered by the cloud are implemented
- Any customer can easily sign in with the cloud provider, enter the credential and billing details, and use the services offered
- offer solutions for minimizing IT infrastructure costs and serve as a viable option for handling peak loads on the local infrastructure
- A fundamental characteristic of public clouds is multi-tenancy
- A public cloud is meant to serve a multitude of users, not a single customer.
- Any customer requires a virtual computing environment that is separated, and most likely isolated, from other users
- QoS management is a very important aspect of public clouds.
- Hence, a significant portion of the software infrastructure is devoted to monitoring the cloud resources, to bill them according to the contract made with the user, and to keep a complete history of cloud usage for each customer
- A public cloud can offer any kind of service: infrastructure, platform, or applications
- Example:

○ Amazon EC2: provides IaaS

○ Google AppEngine is a public cloud that provides an application development PaaS; and

○ SalesForce.com is a public cloud that provides SaaS

- one or more data-centers constitute the physical infrastructure on top of which the services are implemented and delivered.
- Public clouds can be composed of geographically dispersed data-centers to share the load of users and better serve them according to their locations
- Example: AWS has data-centers installed in the United States, Europe, Singapore, and Australia