DIGITAL NOTES
ON

CLOUD COMPUTING
(R22A0522) (R22)
REGULATION

B.TECH IV YEAR – ISEM

(2025-26)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY

(Autonomous Institution – UGC, Govt. of India)

(Affiliated to JNTUH, Hyderabad, Approved by AICTE - Accredited by NBA & NAAC – 'A' Grade - ISO 9001:2015 Certified)Maisammaguda, Dhulapally (Post Via. Hakimpet), Secunderabad – 500100, Telangana State, INDIA.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Vision

To acknowledge quality education and instill high patterns of discipline making the students technologically superior and ethically strong which involves the improvement in the quality of life in human race.

Mission

- To achieve and impart holistic technical education using the best of infrastructure, outstanding technical and teaching expertise to establish the students into competent and confident engineers.
- Evolving the center of excellence through creative and innovative teaching learning practicesforpromotingacademicachievementtoproduceinternationallyacceptedcompetitiveand world class professionals.

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

PEO1–ANALYTICALSKILLS

- To facilitate the graduates with the ability to visualize, gather information, articulate, analyze, solve complex problems, and make decisions. These are essential to address the challenges of complex and computation intensive problems increasing their productivity.

PEO2–TECHNICALSKILLS

- Tofacilitatethegraduateswiththetechnicalskillsthatpreparethemforimmediateemploymentandpursue certification providing a deeper understanding of the technology in advanced areas of computer science and related fields, thus encouraging pursuing higher education and research based on their interest.

PEO3–SOFTSKILLS

- To facilitate the graduates with the soft skills that include fulfilling the mission, setting goals, showing self confidence by communicating effectively, having a positive attitude, get involved in team-work, being a leader, managing their career and their life.

PEO4–PROFESSIONALETHICS

- To facilitate the graduates with the knowledge of professional and ethical responsibilities by paying attention to grooming, being conservative with style, following dress codes, safety codes, and adapting them to technological advancements.

PROGRAM SPECIFIC OUTCOMES (PSOs)

After the completion of the course, B.Tech Computer Science and Engineering, the graduates will have the following Program Specific Outcomes:

1.FundamentalsandcriticalknowledgeoftheComputerSystem:-
AbletoUnderstandtheworkingprinciples of the computer System and its components, Apply the knowledge to build, asses, and analyze the software and hardware aspects of it.

2.The comprehensive and Applicative knowledge of Software Development: Comprehensive skills of Programming Languages, Software process models, methodologies, and able to plan, develop, test, analyze, and manage the software and hardware intensive systems in heterogeneous platforms individually or working in teams.

3.Applications of Computing Domain & Research: Able to use the professional, managerial, interdisciplinary skill set, and domain specific tools in development processes, identify their search gaps, and provide innovative solutions to them.

PROGRAM OUTCOMES (POs)

Engineering Graduates should possess the following:

1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals,andanengineeringspecializationtothesolutionofcomplexengineeringproblems.

2. Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. Design / development of solutions: Design solutions for complex engineeringproblemsanddesignsystemcomponentsorprocessesthatmeetthespecifiedneedswithappropriateconsideration for thepublic health and safety, and the cultural, societal, and environmental considerations.

4. Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and team work: Function effectively as an individual, and as member or leader in diverse teams, and in multidisciplinary settings.

10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

CLOUD COMPUTING

Objectives

- To understand the various distributed system models and evolvingcomputingparadigms
- To gain knowledge in virtualization of computer resources
- To realize the reasons for migrating into cloud
- To introduce the various levels of services that can be achieved by a cloud.
- To describe the security aspects in cloud and the services offered by a cloud.

Unit - I

Cloud Computing Fundamentals: Definition of Cloud computing, Roots of Cloud Computing , Layers and Types of Clouds, Desired Features of a Cloud, Cloud Infrastructure Management, Infrastructure as a Service Providers, Platform as a Service Providers. **Computing Paradigms**: High-Performance Computing, Parallel Computing, Distributed Computing, Cluster Computing, Grid Computing.

UNIT- II

Migrating into a Cloud: Introduction, Broad Approaches to Migrating into the Cloud, the Seven-Step Model of Migration into a Cloud.

Virtualization: Virtual Machines and Virtualization of Clusters and data centers-Implementation Levels of Virtualization -Virtualization Structures/Tools and Mechanisms-Virtualization of CPU, Memory, and I/O Devices-Virtual Clusters and Data Centers

UNIT- III

Infrastructure as a Service (IAAS) & Platform (PAAS): Virtual machines provisioning and Migration services, Virtual Machines Provisioning and Manageability, Virtual Machine Migration Services, VM Provisioning and Migration in Action. On the Management of Virtual machines for Cloud Infrastructures- Aneka—Integration of Private and Public Clouds.

UNIT- IV

Software as a Service (SAAS) &Data Security in the Cloud: Software as a Service SAAS), Google App Engine – Centralizing Email Communications- Collaborating via Web-Based Communication Tools-An Introduction to the idea of Data Security.

UNIT- V

SLA Management in cloud computing: Traditional Approaches to SLO Management, Typesof SLA, Life Cycle of SLA, SLA Management in Cloud.

COURSE OUTCOMES:

1. Ability to analyze various service delivery models of cloud computing
2. Ability to interpret the ways in which the cloud can be programmed and deployed.
3. Ability to comprehend the virtualization and cloud computing concepts
4. Assess the comparative advantages and disadvantages of Virtualization technology
5. Analyze authentication, confidentiality and privacy issues in cloud computing

INDEX

S. No	Name of the Topic	Page No
1	**Cloud Computing Fundamentals**	1
2	Definition of Cloud computing	1
3	Roots of Cloud Computing	2
4	Layers and Types of Clouds	5
5	Desired Features of a Cloud	11
6	Cloud Infrastructure Management	11
7	**Computing Paradigms**	23
8	High-Performance Computing	24
9	Parallel Computing	24
10	Distributed Computing	25
11	Cluster Computing	26
12	Grid Computing	26
13	**Migrating into a Cloud**	28
14	Broad Approaches to Migrating into the Cloud	29
15	Seven-Step Model of Migration into a Cloud	30
16	**Virtualization:** Implementation Levels of Virtualization	31
17	Virtualization Structures/Tools and Mechanisms	35
18	Virtualization of CPU, Memory, and I/O Devices	39
19	Virtual Clusters and Data Centers	44
20	**Infrastructure as a Service (IAAS) & Platform (PAAS)**	47
21	Virtual machines provisioning and Migration services	50
22	Virtual Machines Provisioning and Manageability	56
23	Virtual Machine Migration Services	58
24	VM Provisioning and Migration in Action	58
25	On the Management of Virtual machines for CloudInfrastructures	60
26	Aneka—Integration of Private and Public Clouds.	61
27	**Software as a Service (SAAS) &Data Security in the Cloud**	66
28	Software as a Service SAAS)	66
29	Google App Engine – Centralizing Email Communications	69

30	Collaborating via Web-Based Communication Tools	73
31	**SLA Management in cloud computing**	79
32	Traditional Approaches to SLO Management	79
33	Types of SLA	80
34	Life Cycle of SLA	81
35	SLA Management in Cloud	82

UNIT- I

Cloud Computing Fundamentals: Definition of Cloud computing, Roots of Cloud Computing , Layers and Types of Clouds, Desired Features of a Cloud, Cloud Infrastructure Management, Infrastructure as a Service Providers, Platform as a Service Providers.

Computing Paradigms: High-Performance Computing, Parallel Computing, Distributed Computing, Cluster Computing, Grid Computing.

Introduction to Cloud Computing

"Cloud is a parallel and distributed computing system consisting of a collection ofinter-connected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service- level agreements (SLA) established through negotiation between the service provider and consumers."

"Clouds are a large pool of easily usable and accessible virtualized resources (suchas hardware, development platforms and/or services). These resources can be dynamically reconfigured to adjust to a variable load (scale), allowing also for an optimum resource utilization"

"This pool of resources is typically exploited by a pay-per-use model in which guarantees are offered by the Infrastructure Provider by means of customized Service Level Agreements."

"Clouds are hardware based services offering compute, network, and storage capacity where Hardware management is highly abstracted from the buyer, buyers incur infrastructure costs as variable OPEX, and infrastructure capacityis highly elastic."

Key characteristics of cloud computing

a. the illusion of infinite computing resources;
b. the elimination of an up-front commitment by cloud users;
c. the ability to pay for use…as needed

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

Most common characteristics which a cloud should have:
(i) pay-per-use (no ongoing commitment, utility prices); (ii) elastic capacity and the illusion of infinite resources; (iii) self-service interface; and (iv) resourcesthat is abstracted or virtualized.

Roots of Cloud Computing

The roots of clouds computing can be tracked by observing the advancement of several technologies, especially in hardware (virtualization, multi-core chips), Internet technologies (Web services, service-oriented architectures, Web 2.0),distributed computing (clusters, grids), and systems management (autonomic computing, data center automation).

Figure 1.1 shows the convergence of technology fields that significantly advanced and contributed to the advent of cloud computing.

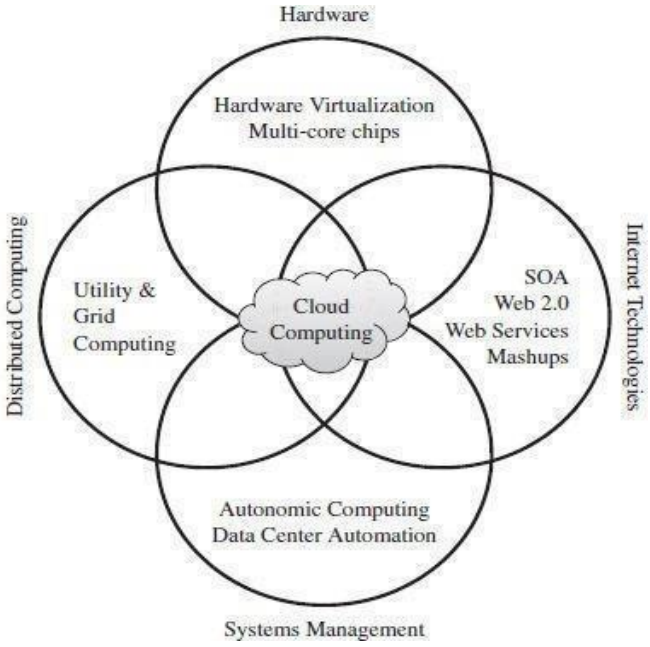


FIGURE 1.1. Convergence of various advances leading to the advent of cloud computing.

The IT world is currently experiencing a switch from in-house generated computing power into utility-supplied computing resources delivered over the Internet as Web services.

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

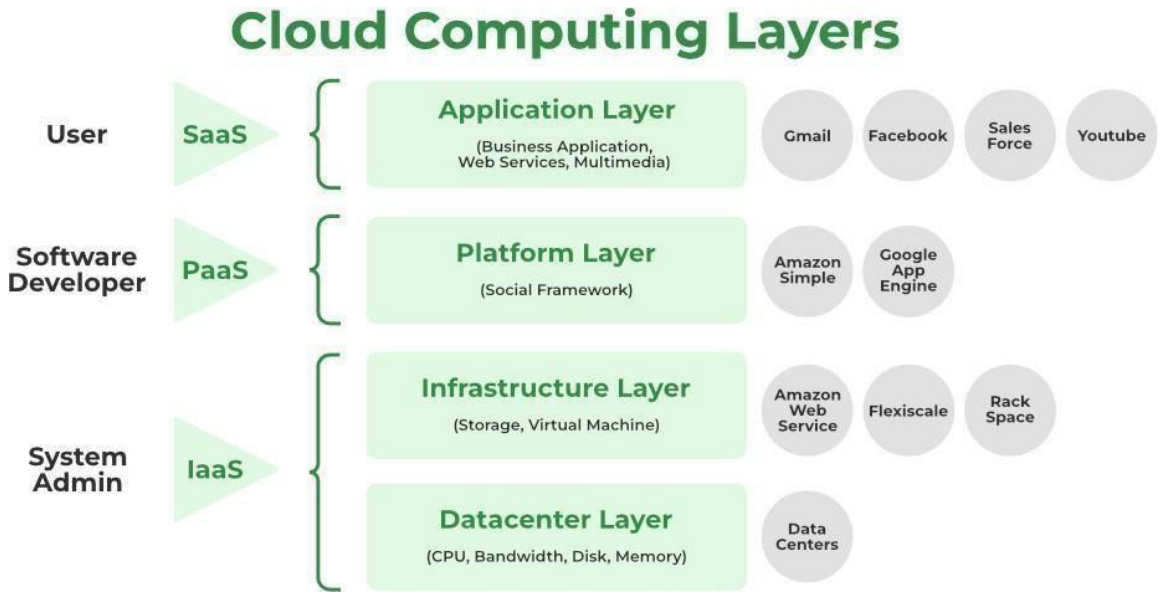
This model brings benefits to both consumers and providers of IT services. Consumers can attain reduction on IT-related costs by choosing to obtain cheaper services from external providers as opposed to heavily investing on IT infrastructure and personnel hiring. The "on-demand" component of this model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

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

In the 1970s, companies who offered common data processing tasks, such as payroll automation, operated time-shared main frames as utilities, which could serve dozens of applications and often operated close to 100% of their capacity.

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

In addition, the unavailability of efficient computer networks meant that IT infrastructure should be hosted in proximity to where it would be consumed. Altogether, these facts have prevented the utility computing reality of taking place on modern computer systems. These facts reveal the potential of delivering computing services with the speed and reliability that businesses enjoy with their local machines. The benefits of economies of scale and high utilization allow providers to offer computing services for a fraction of what it costs for a typical company that generates its own computing power.



SOA, Web Services, Web 2.0, and Mashups

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

WS standards have been created on top of existing ubiquitous technologies such as HTTP and XML, thus providing a common mechanism for delivering services, making them ideal for implementing a service-oriented architecture (SOA). The purpose of a SOA is to address requirements of loosely coupled, standards-based, and protocol-independent distributed computing. In a SOA, software resources arepackaged as “services,” which are well-defined, self-contained modules that provide standard business functionality and are independent of the state or context of other services.

Services are described in a standard definition language and have a published interface. The maturity of WS has enabled the creation of powerful services that can be accessed on-demand, in a uniform way. An enterprise application that follows the SOA paradigm is a collection of services that together perform complex business logic.

In the consumer Web, information and services may be programmatically aggregated, acting as building blocks of complex compositions, called service mashups. Many service providers, such as Amazon, del.icio.us, Facebook, and Google, make their service APIs publicly accessible using standard protocols such as SOAP and REST. Consequently, one can put an idea of a fully functional Web application into practice just by gluing pieces with few lines of code.

In the Software as a Service (SaaS) domain, cloud applications can be built as compositions of other services from the same or different providers. Services such as user authentication, e-mail, payroll management, and calendars are examples of building blocks that can be reused and combined in a business solution in case a single, ready-made system does not provide all those features.

Grid Computing

Grid computing enables aggregation of distributed resources and transparently access to them. Most production grids such as Tera Grid and EGEE seek to share compute and storage resources distributed across different administrative domains, with their main focus being speeding up a broad range of scientific applications, such as climate modeling, drug design, and protein analysis.
A key aspect of the grid vision realization has been building standard Web services-based protocols that allow distributed resources to be "discovered, accessed, allocated, monitored, accounted for, and billed for, etc., and in general managed as a single virtual system." The Open Grid Services Architecture (OGSA) addresses this need for standardization by defining a set of core capabilities and behaviors that address key concerns in grid systems.

Globus Toolkit is a middleware that implements several standard Grid services andover the years has aided the deployment of several service-oriented Grid infrastructures and applications.

The development of standardized protocols for several grid computing activities has contributed—theoretically—to allow delivery of on-demand computing services over the Internet. However, ensuring QoS in grids has been perceived as a difficult endeavor. Lack of performance isolation has prevented grids adoption in a variety of scenarios, especially on environments where resources are oversubscribed or users are uncooperative.

Another issue that has lead to frustration when using grids is the availability of resources with diverse software configurations, including disparate operating systems, libraries, compilers, runtime environments, and so forth. At the same time, user applications would often run only on specially customized environments. Consequently, a portability barrier has often been present on most grid infrastructures, inhibiting users of adopting grids as utility computing environments

Utility Computing

In utility computing environments, users assign a "utility" value to their jobs, where utility is a fixed or time-varying valuation that captures various QoS constraints (deadline, importance, satisfaction). The valuation is the amount they are willing to pay a service provider to satisfy their demands. The service providersthen attempt to maximize their own utility, where said
utility may directly correlate with their profit. Providers can choose to prioritize high yield (i.e., profit per unit of resource) user jobs, leading to a scenario where shared systems are viewed as a marketplace, where users compete for resources based on the perceived utility or value of their jobs.

Hardware Virtualization

Cloud computing services are usually backed by large-scale data centers composed of thousands of computers. Such data centers are built to serve many users and host many disparate applications.

The idea of virtualizing a computer system's resources, including processors, memory, and I/O devices, has been well established for decades, aiming at improving sharing and utilization of computer systems. Hardware virtualization allows running multiple operating systems and software stacks on a single physicalplatform.

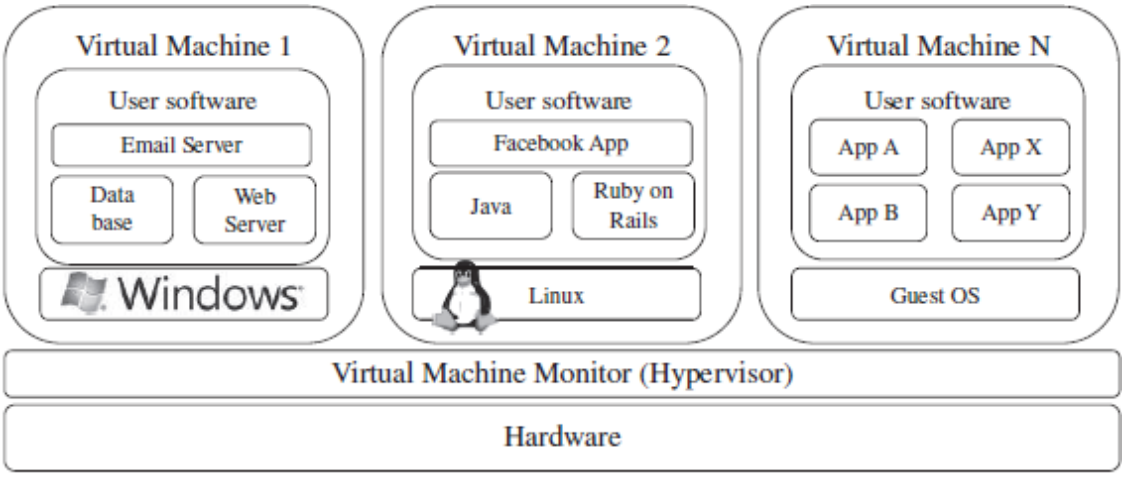


FIGURE 1.2. A hardware virtualized server hosting three virtual machines, each one running distinct operating system and user level software stack.

As depicted in Figure 1.2, a software layer, the virtual machine monitor (VMM), also called a hypervisor, mediates access to the physical hardware presenting to each guest operating system a virtual machine(VM), which is a set of virtual platform interfaces

The advent of several innovative technologies—multi-core chips, para- virtualization, hardware-assisted virtualization, and live migration of VMs—has contributed to an increasing adoption of virtualization on server systems.

Perceived benefits were improvements on sharing and utilization, better manageability, and higher reliability.

There are three basic capabilities regarding management of workload in a virtualized system, namely isolation, consolidation, and migration

Workload isolation is achieved since all program instructions are fully confined inside a VM, which leads to improvements in security. Better reliability is also achieved because software failures inside one VM do not affect others

The consolidation of several individual and heterogeneous workloads onto a single physical platform leads to better system utilization. This practice is also employed for overcoming potential software and hardware incompatibilities incase of

upgrades, given that it is possible to run legacy and new operation systems concurrently

Workload migration, also referred to as application mobility, targets at facilitating hardware maintenance, load balancing, and disaster recovery. It is done by encapsulating a guest OS state within a VM and allowing it to be suspended, fully serialized, migrated to a different platform, and resumed immediately or preserved to be restored at a later date. A VM's state includes a full disk or partition image, configuration files, and an image of its RAM.

A number of VMM platforms exist that are the basis of many utility or cloud computing environments. The most notable ones are VMWare, Xen, andKVM.
VMWARE ESXi: is a VMM from VMWare. It is a bare-metal hypervisor,meaning that it installs directly on the physical server, whereas others may requirea host operating system. It provides advanced virtualization techniques ofprocessor, memory, and I/O. Especially, through page sharing, it can over commitmemory, thus increasing the density of VMs inside a single physical server.

Xen : The Xen hypervisor started as an open-source project and has served as a base to other virtualization products, both commercial and open-source. It has pioneered the para-virtualization concept, on which the guest operating system, by means of a specialized kernel, can interact with the hypervisor, thus significantly improving performance.

KVM: The kernel-based virtual machine (KVM) is a Linux virtualization subsystem. It has been part of the mainline Linux kernel since version 2.6.20, thus being natively supported by several distributions. In addition, activities such as memory management and scheduling are carried out by existing kernel features, thus making KVM simpler and smaller than hypervisors that take control of the entire machine

Virtual Appliances and the Open Virtualization Format

An application combined with the environment needed to run it (operating system, libraries, compilers, databases, application containers, and so forth) is referred toas a "virtual appliance."

Packaging application environments in the shape of virtual appliances eases software customization, configuration, and patching and improves portability. Most commonly, an appliance is shaped asa VM disk image associated with hardware requirements, and it can be readily deployed in a hypervisor. The VMWare virtual appliance marketplace allows users to deploy appliances onVMWare hypervisors or on partners public clouds, and Amazon allows developersto share specialized Amazon Machine Images (AMI) and monetize their usage on Amazon EC2.

In a multitude of hypervisors, where each one supports a different VM image format and the formats are incompatible with one another, a great deal of interoperability issues arises. In order to facilitate packing and distribution of software to be run on VMs several vendors, including VMware, IBM, Citrix, Cisco, Microsoft, Dell, and HP, have devised the Open Virtualization Format

(OVF). It aims at being "open, secure, portable, efficient and extensible" [32]. An OVF package consists of a file, or set of files, describing the VM hardware characteristics (e.g., memory, network cards, and disks), operating system details, startup, and shutdown actions, the virtual disks themselves, and other metadata containing product and licensing information. OVF also supports complex packages composed of multiple VMs.

Autonomic Computing

The increasing complexity of computing systems has motivated research on autonomic computing, which seeks to improve systems by decreasing human involvement in their operation. In other words, systems should manage themselves,with high-level guidance from humans

Autonomic, or self-managing, systems rely on monitoring probes and gauges (sensors), on an adaptation engine (autonomic manager) for computing optimizations based on monitoring data, and on effectors to carry out changes on the system. IBM's Autonomic Computing Initiative has
contributed to define the four properties of autonomic systems: self-configuration, self-optimization, self-healing, and self-protection. IBM has also suggested a reference model for autonomic control loops of autonomic managers, called MAPE-K (Monitor Analyze Plan Execute—Knowledge)

The large data centers of cloud computing providers must be managed in anefficient way. In this sense, the concepts of autonomic computing inspiresoftware technologies for data center automation, which may perform taskssuch as: management of service levels of running applications; management ofdata center capacity; proactive disaster recovery; and automation of VMprovisioning

LAYERS AND TYPES OFCLOUDS

Cloud computing services are divided into three classes
(1) Infrastructure as a Service, (2) Platform as a Service, and (3) Softwareas aService

Figure 1.3 depicts the layered organization of the cloud stackfrom physical infrastructure to applications.

These abstraction levels can also be viewed as a layered architecture whereservices of a higher layer can be composed from services of the underlying layerA core middleware manages physical resources andthe VMs deployed on top of them; in addition, it provides the required features(e.g., accounting and billing) to offer multi-tenant pay- as-you-goservices.

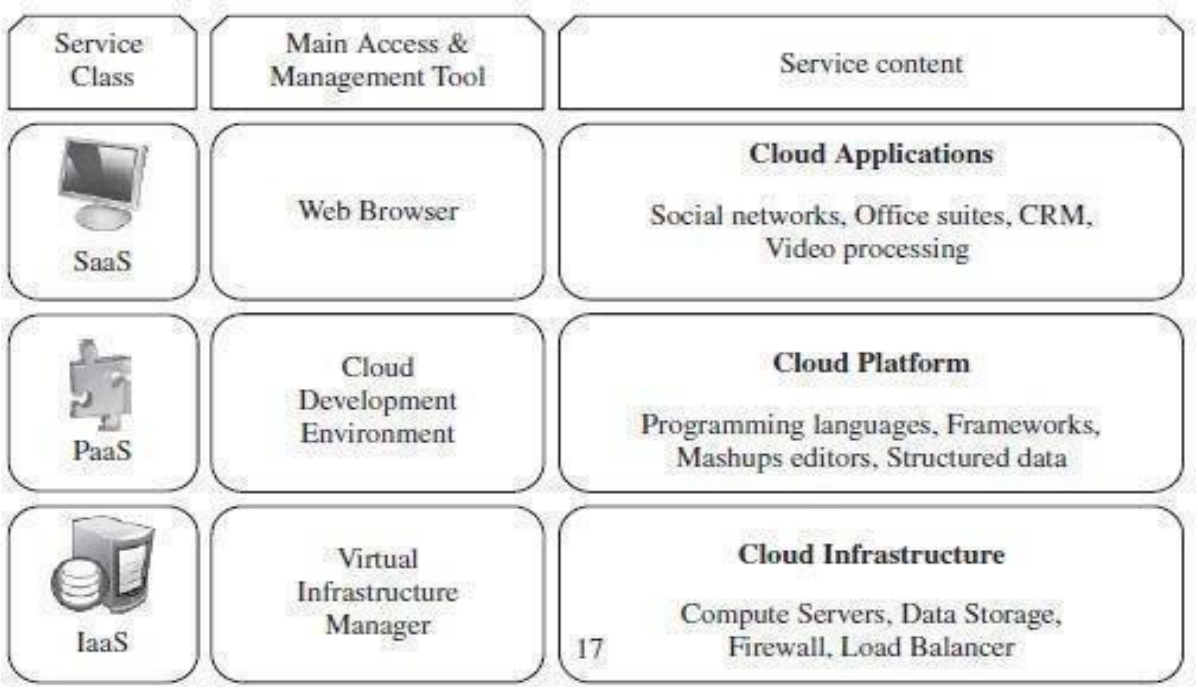


FIGURE 1.3 The cloud computing stack Infrastructure as a Service Offering virtualized resources (computation, storage, and communication) on demand is known as Infrastructure as a Service (IaaS). A cloud infrastructure enables on-demand provisioning of servers running several choices of operating systems and a customized software stack. Infrastructure services are considered to be the bottom layer of cloud computing systems.

Amazon Web Services mainly offers IaaS, which in the case of its EC2service means offering VMs with a software stack that can be customized similar to how an ordinary physical server would be customized. Users are given privileges to perform numerous activities to the server, such as: starting and stopping it, customizing it by installing software packages, attaching virtual disks to it, and configuring access permissions and firewalls rules.

Platform as a Service

A cloud platform offers an environment on which developers create and deploy applications and do not necessarily need to know how many processors or how much memory that applications will be using. In addition, multiple programming models and specialized services (e.g., data access, authentication, and payments) are offered as building blocks to new applications.

Google App Engine, an example of Platform as a Service, offers a scalable environment for developing and hosting Web applications, which should be written in specific programming languages such as Python or Java, and use the services' own proprietary structured object data store.

Software as a Service

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

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

Deployment Models

A cloud can be classified as public, private, community, or hybrid based on model of deployment as shown in Figure 1.4.

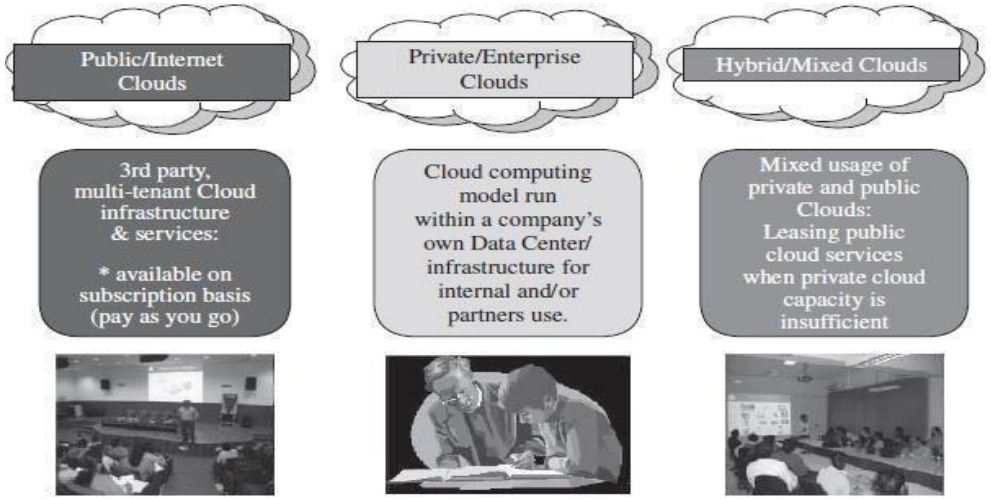


FIGURE 1.4. Types of clouds based on deployment models.

Public cloud: "cloud made available in a pay-as-you-go manner to the general public"
Private cloud: "internal data center of a business or other organization, not made available to the general public."
Community cloud:"shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations) **Hybrid cloud** takes shape when a private cloud is supplemented with computing capacity from public clouds.
The approach of temporarily renting capacity to handle spikes in load is known as **"cloud- bursting"**

Features of a Cloud

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

Self-Service: clouds must allow self-service access so that customers can request, customize, pay, and use services without intervention of human operators

Per-Usage Metering and Billing: Cloud computing eliminates up-front commitment by users, allowing them to request and use only the necessary amount. Services must be priced on a short term basis (e.g., by the hour), allowingusers to release (and not pay for) resources as soon as they are not needed

Elasticity: Cloud computing gives the illusion of infinite computing resources available on demand. Therefore users expect clouds to rapidly provide resources inany quantity at any time. In particular, it is expected that the additional resources can be (a) provisioned, possibly automatically, when an application load increases and (b) released when load decreases (scale up and down)

Customization: resources rented from the cloud must be highly customizable. customization means allowing users to deploy specialized virtual appliances and to be given privileged (root) access to the virtual servers.

CLOUD INFRASTRUCTURE MANAGEMENT

A key challenge IaaS providers face when building a cloud infrastructure is managing physical and virtual resources, namely servers, storage, and networks, in a holistic fashion. The orchestration of resources must be performed in a way to rapidly and dynamically provision resources to applications.

The software toolkit responsible for this orchestration is called a virtual infrastructure manager (VIM). This type of software resembles a traditional operating system—but instead of dealing with a single computer, it aggregates resources from multiple computers, presenting a uniform view to user and applications.

Features

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

Self-Service, On-Demand Resource Provisioning: Self-service access to resources has been perceived as one the most attractive features of clouds. This feature enables users to directly obtain services from clouds, such as spawning the creation of a server and tailoring its software, configurations, and security policies,

without interacting with a human system administrator. This capability "eliminatesthe need for more time-consuming, labor-intensive, human driven procurement processesfamiliar to many in IT".

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

Storage Virtualization: Virtualizing storage means abstracting logical storage from physical storage. By consolidating all available storage devices in a data center, it allows creating virtual disks independent from device and location.

Interface to Public Clouds: Extending the capacity of a local in-house computing infrastructure by borrowing resources from public clouds is advantageous. In this fashion, institutions can make good use of their available resources and, in case of spikes in demand, extra load can be offloaded to rented resources. A VI manager can be used in a hybrid cloud setup if it offers a driver to manage the life cycle of virtualized resources obtained from external cloud providers. To the applications, the use of leased resources must ideally be transparent.

Virtual Networking: Virtual networks allow creating an isolated network on top of a physical infrastructure independently from physical topology and locations. A virtual LAN (VLAN) allows isolating traffic that shares a switched network, allowing VMs to be grouped into the same broadcast domain. Additionally, a VLAN can be configured to block traffic originated from VMs from other networks.

Dynamic Resource Allocation: Increased awareness of energy consumption in data centers has encouraged the practice of dynamic consolidating VMs in a fewer number of servers. In cloud infrastructures, where applications have variable and dynamic needs, capacity management and demand prediction are especially complicated. This fact triggers the need for dynamic resource allocation aiming at obtaining a timely match of supply and demand. Energy consumption reduction and better management of SLAs can be achieved by dynamically remapping VMs to physical machines at regular intervals. Machines that are not assigned any VM can be turned off or put on a low power state. In the same fashion, overheating canbe avoided by moving load away from hotspots.

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

Reservation and Negotiation Mechanism: When users request computational resources to available at a specific time, requests are termed advance reservations

(AR), in contrast to best- effort requests, when users request resources whenever available. To support complex requests, such as AR, a VI manager must allow users to "lease" resources expressing more complex terms (e.g., the period of time of a reservation). This is especially useful in clouds on which resources are scarce; since not all requests may be satisfied immediately, they can benefit of VM placement strategies that support queues, priorities, and advance reservations. Additionally, leases may be negotiated and renegotiated, allowing provider and consumer to modify a lease or present counter proposals until an agreement is reached. This feature is illustrated by the case in which an AR request for a given slot cannot be satisfied, but the provider can offer a distinct slot that is still satisfactory to the user. This problem has been addressed in OpenPEX, which incorporates a bilateral negotiation protocol that allows users and providers to come to an alternative agreement by exchanging offers and counter offers.

High Availability and Data Recovery: The high availability (HA) feature of VI managers aims at minimizing application downtime and preventing business disruption. A few VI managers accomplish this by providing a failover mechanism, which detects failure of both physical and virtual servers and restarts VMs on healthy physical servers. This style of HA protects from host, but not VM, failures. For mission critical applications, when a failover solution involving restarting VMs does not suffice, additional levels of fault tolerance that rely on redundancy of VMs are implemented. In this style, redundant and synchronized VMs (running or in standby) are kept in a secondary physical server. The HA solution monitors failures of system components such as servers, VMs, disks, and network and ensures that a duplicate VM serves the application in case of failures. Data backup in clouds should take into account the high data volume involved in VM management. Frequent backup of a large number of VMs, each one with multiple virtual disks attached, should be done with minimal interference in the systems performance. In this sense, some VI managers offer data protection mechanisms that perform incremental backups of VM images.
The backup workload is often assigned to proxies, thus offloading production server and reducing network overhead

Case Studies

Apache VCL: The Virtual Computing Lab project has been incepted in 2004 by researchers at the North Carolina State University as a way to provide customized environments to computer lab users. Apache VCL provides the following features:
(i) multi-platform controller, based on Apache/PHP;(ii)WebportalandXML-RPCinterfaces;(iii)supportforVMwarehypervisors (ESX, ESXi, and Server); (iv) virtual networks; (v) virtual clusters; and (vi)advance reservation of capacity.

AppLogic.AppLogic: is a commercial VI manager, the flagship product of 3tera Inc. from California, USA. The company has labeled this product as a Grid Operating System.
AppLogic provides the following features: Linux-based controller; CLI and GUI interfaces; Xen backend; Global Volume Store (GVS) storage virtualization; virtual networks; virtual clusters; dynamic resource allocation; high availability; and data protection.

Citrix Essentials: The Citrix Essentials suite is one the most feature complete VI management software available, focusing on management and automation of data centers. It is essentially a hypervisor-agnostic solution, currently supporting Citrix XenServer and Microsoft Hyper-V. Citrix Essentials provides the following features: Windowsbased controller; GUI, CLI, Web portal, and XML-RPC interfaces; support for XenServer and Hyper-V hypervisors; Citrix Storage Link storage virtualization;virtual networks; dynamic resource allocation; three-level high availability (i.e., recovery by VM restart, recovery by activating paused duplicate VM, and running duplicate VM continuously); data protection with Citrix ConsolidatedBackup.

Enomaly ECP: The Enomaly Elastic Computing Platform, in its most complete edition, offers most features a service provider needs to build an IaaS cloud.
Enomaly ECP provides the following features: Linux-based controller; Web portal and Web services (REST) interfaces; Xen back-end; interface to the Amazon EC2 public cloud; virtual networks; virtual clusters (ElasticValet) Eucalyptus.

The Eucalyptus: framework was one of the first open-source projects to focus on building IaaS clouds. It has been developed with the intent of providing an open-source implementation nearly identical in functionality to Amazon Web ServicesAPIs.
Eucalyptus provides the following features: Linux-based controller with administration Web portal; EC2-compatible (SOAP, Query) and S3-compatible (SOAP, REST) CLI and Web portal interfaces; Xen, KVM, and VMWare backends; Amazon EBS-compatible virtual storage devices; interface to the Amazon EC2 public cloud; virtualnetworks.

Nimbus3: The Nimbus toolkit is built on top of the Globus framework. Nimbus provides most features in common with other open-source VI managers, such as an EC2-compatible front-end API, support to Xen, and a backend interface to Amazon EC2. However, it distinguishes from others by providing a Globus Web Services Resource Framework (WSRF) interface. It also provides a backend service, named Pilot, which spawns VMs on clusters managed by a local resource manager (LRM) such as PBS and SGE.

OpenNebula: OpenNebula is one of the most feature-rich open-source VI managers. It was initially conceived to manage local virtual infrastructure, but has also included remote interfaces that make it viable to build public clouds. Altogether, four programming APIs are available: XML-RPC and libvirt for local interaction; a subset of EC2 (Query) APIs and the OpenNebula Cloud API (OCA) for public access. OpenNebula provides the following features: Linux-based controller; CLI, XML-RPC, EC2-compatible Query and OCA interfaces; Xen, KVM, and VMware backend; interface to public clouds (Amazon EC2, ElasticHosts); virtual networks; dynamic resource allocation; advance reservation of capacity.

OpenPEX: OpenPEX (Open Provisioning and EXecution Environment) was constructed around the notion of using advance reservations as the primary methodfor allocatingVMinstances.
OpenPEX provides the following features: multi-platform (Java) controller; Web

portal and Web services (REST) interfaces; Citrix XenServer backend; advance reservation of capacity with negotiation.

oVirt: oVirt is an open-source VI manager, sponsored by Red Hat's Emergent Technology group oVirt provides the following features: Fedora Linux-based controller packaged as a virtual appliance; Web portal interface; KVMbackend.

Platform ISF: Infrastructure Sharing Facility (ISF) is the VI manager offering from Platform Computing. The company, mainly through its LSF family of products, has been serving the HPC market for several years ISF provides the following features: Linux-based controller packaged as a virtual appliance; Web portal interface; dynamic resource allocation; advance reservation of capacity; high availability.

VMWare vSphere and vCloud: vSphere is VMware's suite of tools aimed at transforming IT infrastructures into private clouds. In the vSphere architecture, servers run on the ESXi platform. A separate server runs vCenter Server, which centralizes control over the entire virtual infrastructure. Through the vSphere Client software, administrators connect to vCenter Server to perform various tasks. The Distributed Resource Scheduler (DRS) makes allocation decisions based on predefined rules and policies. It continuously monitors the amount of resources available to VMs and, if necessary, makes allocation changes to meet VM requirements. In the storage virtualization realm, vStorage VMFS is a cluster file system to provide aggregate several disks in a single volume. VMFS is especially optimized to store VM images and virtual disks. It supports storage equipment that use Fibre Channel or iSCSI SAN. vSphere provides the following features: Windows-based controller (vCenter Server); CLI, GUI, Web portal, and Web services interfaces; VMware ESX, ESXi backend; VMware vStorage VMFS storage virtualization; interface to external clouds (VMware vCloud partners); virtual networks (VMWare Distributed Switch); dynamic resource allocation (VMware DRM); high availability; data protection (VMWare ConsolidatedBackup).

INFRASTRUCTURE AS A SERVICE PROVIDERS

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

Features

IaaS offerings can be distinguished by the availability of specialized features that influence the cost_benefit ratio to be experienced by user applications when movedto the cloud. The most relevant features are: (i) geographic distribution of data centers; (ii) variety of user interfaces and APIs to access the system; (iii) specialized components and services that aid particular applications (e.g., loadbalancers, firewalls); (iv) choice of virtualization platform and operating

systems; and (v) different billing methods and period (e.g., prepaid vs. post-paid, hourly
vs. monthly).

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

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

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

Automatic Scaling and Load Balancing: Elasticity is a key characteristic of the cloud computing model. Applications often need to scale up and down to meet varying load conditions. Automatic scaling is a highly desirable feature of IaaS clouds. It allow users to set conditions for when they want their applications to scale up and down, based on application-specific metrics such as transactions per second, number of simultaneous users, request latency, and so forth. When the number of virtual servers is increased by automatic scaling, incoming traffic must be automatically distributed among the available servers. This activity enables applications to promptly respond to traffic increase while also achieving greater fault tolerance.

Service-Level Agreement: Service-level agreements (SLAs) are offered by IaaS providers to express their commitment to delivery of a certain QoS. To customers it serves as a warranty. An SLA usually include availability and performance guarantees. Additionally, metrics must be agreed upon by all parties as well as penalties for violating these expectations. Most IaaS providers focus their SLA terms on availability guarantees, specifying the minimum percentage of time the system will be available during a certain period.

Hypervisor and Operating System Choice: Traditionally, IaaS offerings have been based on heavily customized open-source Xen deployments. IaaS providers needed expertise in Linux, networking, virtualization, metering, resource management, and many other low-level aspects to successfully deploy and maintain their cloud offerings.

Case Studies

Amazon Web Services: Amazon WS4 (AWS) is one of the major players in the cloud computing market. It pioneered the introduction of IaaS clouds in 2006. It offers a variety cloud services, most notably: S3 (storage), EC2 (virtual servers), Cloudfront (content delivery), Cloudfront Streaming (video streaming), SimpleDB (structured datastore), RDS (Relational Database), SQS (reliable messaging), and Elastic MapReduce (data processing). The Elastic Compute Cloud (EC2) offersXen-based virtual servers (instances) that can be instantiated from Amazon Machine Images (AMIs). Instances are available in a variety of sizes, operating systems, architectures, and price. CPU capacity of instances is measured in Amazon Compute Units and, although fixed for each instance, vary among instance types from 1 (small instance) to 20 (high CPU instance). Each instanceprovides a certain amount of nonpersistent disk space; a persistence disk service (Elastic Block Storage) allows attaching virtual disks to instances with space up to1TB. Elasticity can be achieved by combining the CloudWatch, Auto Scaling,and Elastic Load Balancing features, which allow the number of instances to scale up and down automatically based on a set of customizable rules, and traffic to be distributed across available instances. Fixed IP address (Elastic IPs) are not available by default, but can be obtained at an additional cost.

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

Joyent: Joyent's Public Cloud offers servers based on Solaris containers virtualization technology. These servers, dubbed accelerators, allow deploying various specialized software- stack based on a customized version of Open- Solaris operating system, which include by default a Web-based configuration tool and several pre-installed software, such as Apache, MySQL, PHP, Ruby on Rails, and Java. Software load balancing is available as an accelerator in addition to hardware load balancers. A notable feature of Joyent's virtual servers is automatic vertical scaling of CPU cores, which means a virtual server can make use of additional CPUs automatically up to the maximum number of cores available in the physical host.

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

GoGrid: GoGrid, like many other IaaS providers, allows its customers to utilize a range of pre- made Windows and Linux images, in a range of fixed instance sizes. GoGrid also offers "value- added" stacks on top for applications such as high-volume Web serving, e-Commerce, and database stores. It offers some notable features, such as a "hybrid hosting" facility, which combines traditional dedicated

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

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

PLATFORM AS A SERVICE PROVIDERS

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

Features

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

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

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

Persistence Options: A persistence layer is essential to allow applications to record their state and recover it in case of crashes, as well as to store user data. Web and enterprise application developers have chosen relational databases as the preferred persistence method. These databases offer fast and reliable structured data storage and transaction processing, but may lack scalability to handle several petabytes of data stored in commodity computers. In the cloud computing domain, distributed storage technologies have emerged, which seek to be robust and highly scalable, at the expense of relational structure and convenient query languages.

Case Studies

Aneka: Aneka is a .NET-based service-oriented resource management and development platform. Each server in an Aneka deployment (dubbed Aneka cloud node) hosts the Aneka container, which provides the base infrastructure that consists of services for persistence, security (authorization, authentication and auditing), and communication (message handling and dispatching). Cloud nodes can be either physical server, virtual machines (XenServer and VMware are supported), and instances rented from Amazon EC2. The Aneka container can also host any number of optional services that can be added by developers to augment the capabilities of an Aneka Cloud node, thus providing a single, extensible framework for orchestrating various application models.
Several programming models are supported by such task models to enable execution of legacy HPC applications and MapReduce, which enables a variety of data-mining and search applications. Users request resources via a client to a reservation services manager of the Aneka master node, which manages all cloud nodes and contains scheduling service to distribute request to cloud nodes.

App Engine: Google App Engine lets you run your Python and Java Web applications on elastic infrastructure supplied by Google. App Engine allows your applications to scale dynamically as your traffic and data storage requirements increase or decrease. It gives developers a choice between a Python stack and Java. The App Engine serving architecture is notable in that it allows real-time auto-scaling without virtualization for many common types of Web applications. However, such auto-scaling is dependent on the application developer using a limited subset of the native APIs on each platform, and in some instances you need to use specific Google APIs such as URLFetch, Datastore, and memcache in place of certain native API calls. For example, a deployed App Engine application cannot write to the file system directly (you must use the Google Datastore) or open a socket or access another host directly (you must use Google URL fetch service). A Java application cannot create a new Thread either.
Microsoft Azure: Microsoft Azure Cloud Services offers developers a hosted. NET Stack (C#, VB.Net, ASP.NET). In addition, a Java & Ruby SDK for .NET Services is also available. The Azure system consists of a number of elements. The Windows Azure Fabric Controller provides auto-scaling and reliability, and it manages memory resources and load balancing. The .NET Service Bus registers and connects applications together. The .NET Access Control identity providers include enterprise directories and Windows LiveID. Finally, the .NET Workflow allows construction and execution of workflow instances.

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

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

CHALLENGES AND RISKS

Despite the initial success and popularity of the cloud computing paradigm and the extensive availability of providers and tools, a significant number of challenges and risks are inherent to this new model of computing. Providers, developers, and end users must consider these challenges and risks to take good advantage of cloud computing. Issues to be faced include user privacy, data security, data lock-in, availability of service, disaster recovery, performance, scalability, energy-efficiency, and programmability.

Security, Privacy, and Trust: Security and privacy affect the entire cloud computing stack, since there is a massive use of third-party services and infrastructures that are used to host important data or to perform critical operations. In this scenario, the trust toward providers is fundamental to ensure the desired level of privacy for applications hosted in the cloud. Legal and regulatory issues also need attention. When data are moved into the Cloud, providers may choose to locate them anywhere on the planet. The physical location of data centers determines the set of laws that can be applied to the management of data. For example, specific cryptography techniques could not be used because they are not allowed in some countries. Similarly, country laws can impose that sensitive data, such as patient health records, are to be stored within national borders.

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

The answer to this concern is standardization. In this direction, there are efforts to create open standards for cloud computing. The Cloud Computing Interoperability Forum (CCIF) was formed by organizations such as Intel, Sun, and Cisco in order to "enable a global cloud computing ecosystem whereby organizations are able to seamlessly work together for the purposes for wider industry adoption of cloud computing technology." The development of the Unified Cloud Interface (UCI) by CCIF aims at creating a standard programmatic point of access to an entire cloud infrastructure. In the hardware virtualization sphere, the Open Virtual Format (OVF) aims at facilitating packing and distribution of software to be run on VMs so that virtual appliances can be made portable—that is, seamlessly run on hypervisor of different vendors.

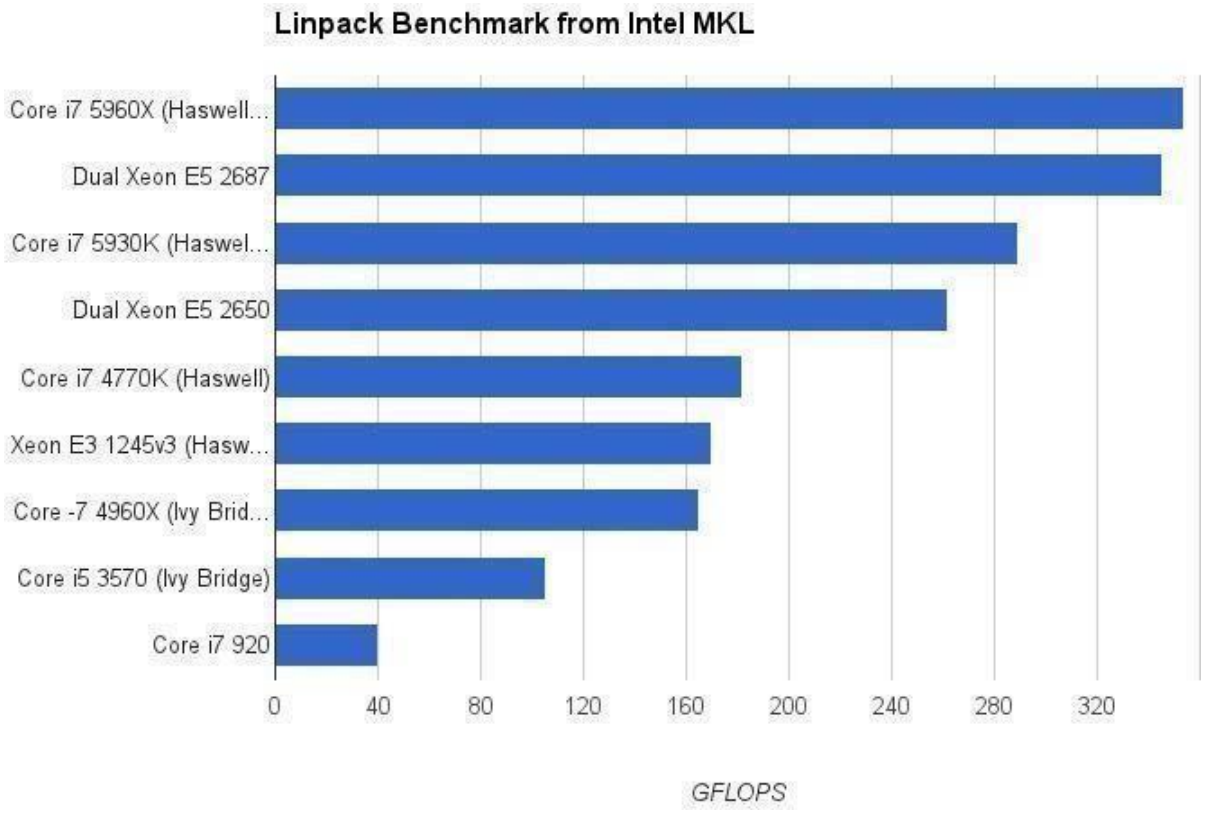
Availability, Fault-Tolerance, and Disaster Recovery: It is expected that users will have certain expectations about the service level to be provided once their applications are moved to the cloud. These expectations include availability of the service, its overall performance, and what measures are to be taken when something goes wrong in the system or its components. In summary, users seek for a warranty before they can comfortably move their business to the cloud. SLAs, which include QoS requirements, must be ideally set up between customers and cloud computing providers to act as warranty. An SLA specifies the details of the service to be provided, including availability and performance guarantees. Additionally, metrics must be agreed upon by all parties, and penalties for violating the expectations must also be approved.

Resource Management and Energy-Efficiency: One important challenge faced by providers of cloud computing services is the efficient management of virtualized resource pools. Physical resources such as CPU cores, disk space, and network bandwidth must be sliced and shared among virtual machines running potentially heterogeneous workloads. The multi-dimensional nature of virtual machines complicates the activity of finding a good mapping of VMs onto available physical hosts while maximizing user utility. Dimensions to be considered include: number of CPUs, amount of memory, size of virtual disks, and network bandwidth. Dynamic VM mapping policies may leverage the ability to suspend, migrate, and resume VMs as an easy way of preempting low-priority allocations in favor of higher-priority ones. Migration of VMs also brings additional challenges such as detecting when to initiate a migration, which VM to migrate, and where to migrate. In addition, policies may take advantage of live migration of virtual machines to relocate data center load without significantly disrupting running services. In this case, an additional concern is the trade-off between the negative impact of a live migration on the performance and stability of a service and the benefits to be achieved with that migration. Another challenge concerns the outstanding amount of data to be managed in various VM management activities. Such data amount is a result of particular abilities of virtual machines, including the ability of traveling through space (i.e., migration) and time (i.e., checkpointing and rewinding), operations that may be required in load balancing, backup, and recovery scenarios. In addition, dynamic provisioning of new VMs and replicating existing VMs require efficient mechanisms to make VM block storage devices (e.g., image files) quickly available at selected hosts. Data centers consumer large amounts of electricity. According to a data published byHP[4], 100 server racks can consume 1.3MWof power and another 1.3 MW are required

by the cooling system, thus costing USD 2.6 million per year. Besides the monetary cost, data centers significantly impact the environment in terms of CO2 emissions from the cooling systems.

High-Performance Computing

- For many years, HPC systems emphasize the raw speed performance.
- The speed of HPC systems has increased from Gflops in the early 1990s to now Pflops in 2010.
- This improvement was driven mainly by the demands from scientific, engineering, and manufacturing communities.
- Top 500 most powerful computer systems in the world are measured by floating-point speed in Linpack benchmark results.
- However, the number of supercomputer users is limited to less than 10% of all computer users.
- Today, the majority of computer users are using desktop computers or large servers when they conduct Internet searches and market-driven computing tasks.



High-Throughput Computing

- The development of market-oriented high-end computing systems is undergoing a strategic change from an **HPC paradigm to an HTC paradigm.**
- This HTC paradigm pays more attention to **high-flux computing.**
- The main application for high-flux computing is in Internet searches and web services by millions or more users simultaneously.
- The performance goal thus shifts to measure high throughput or the **number of tasks completed per unit of time.**

Three New Computing Paradigms

- With the introduction of SOA, **Web 2.0 services** become available.
- Advances in virtualization make it possible to see the growth of **Internet clouds** as a new computing paradigm.
- The maturity of radio-frequency identification (RFID), Global Positioning System (GPS), and sensor technologies has triggered the development of the **Internet of Things (IoT).**

Computing Paradigm Distinctions

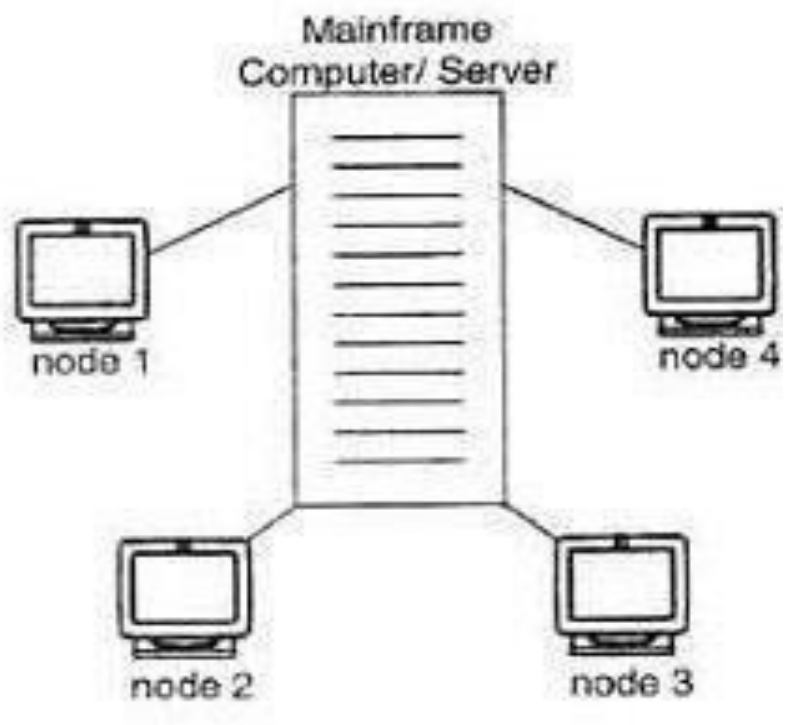
- Centralized computing
- Parallel computing
- Distributed computing
- Cloud computing

In general,

- distributed computing is the opposite of centralized computing.
- The field of parallel computing overlaps with distributed computing to a great extent.
- Cloud computing overlaps with distributed, centralized, and parallel computing.

Centralized computing:

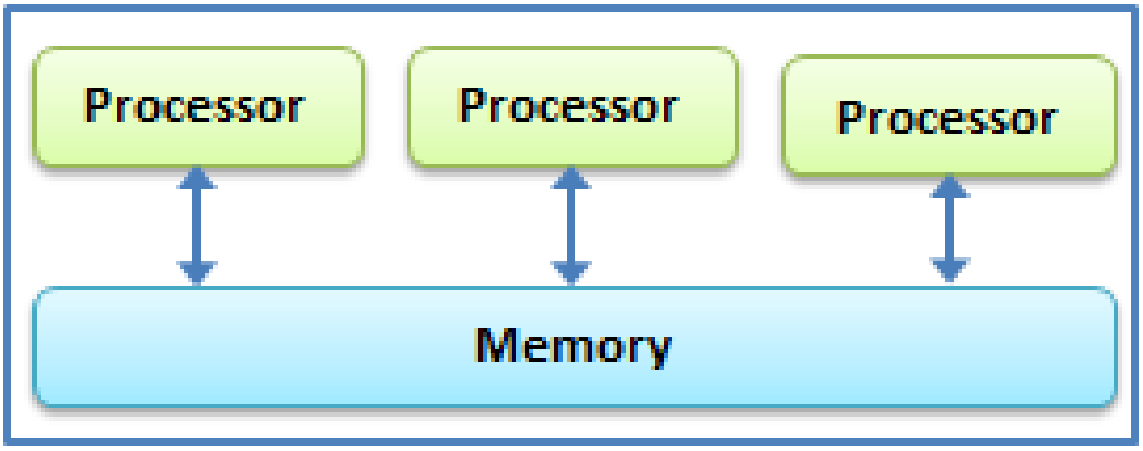
- This is a computing paradigm by which all computer **resources are centralized in one physical system.**
- All resources (processors, memory, and storage) are fully shared and tightly coupled within one integrated OS.
- Many data centers and supercomputers are centralized systems, but they are used in parallel, distributed, and cloud computing applications



Parallel computing:

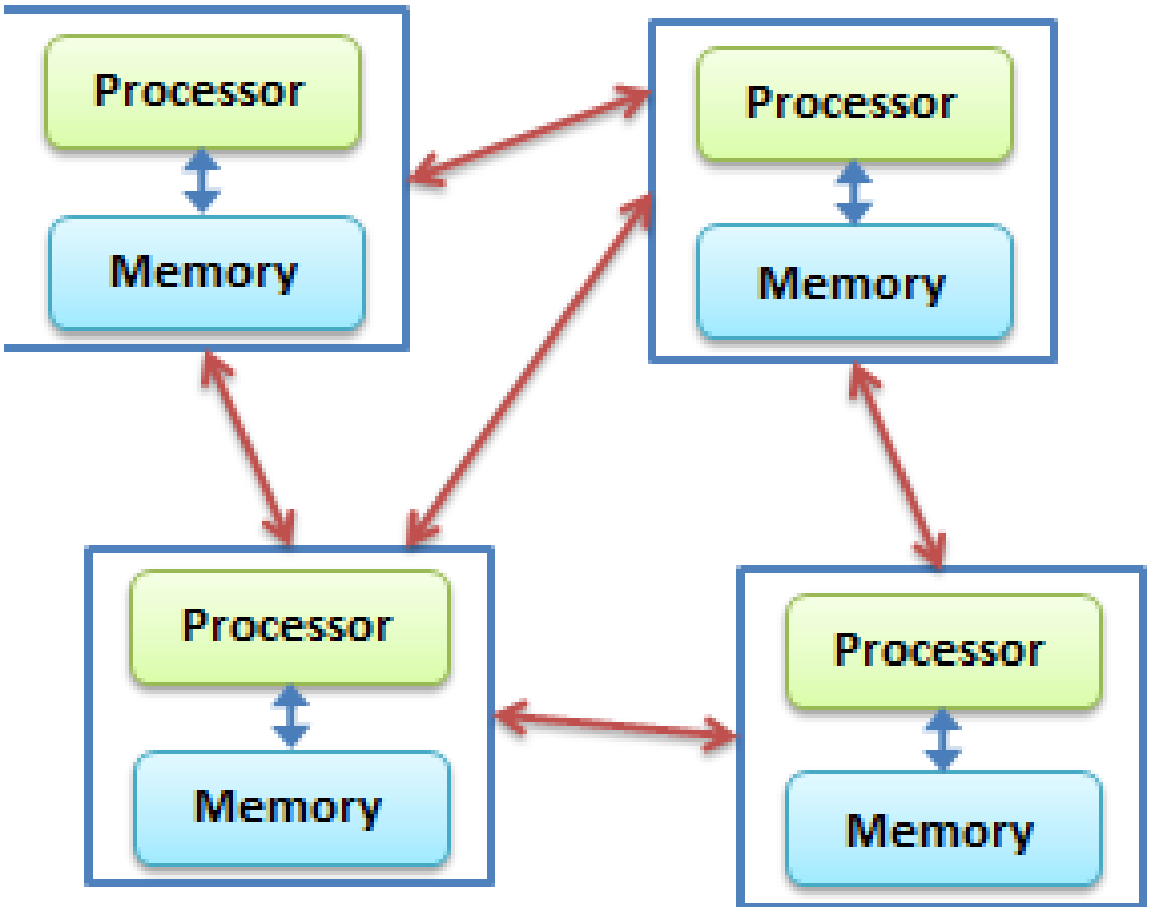
- In parallel computing, all processors are either tightly coupled with centralized shared memory or loosely coupled with distributed memory.
- Interprocessor communication is accomplished through shared memory or via message passing.
- A computer system capable of parallel computing is commonly known as a **parallel computer.**
- Programs running in a parallel computer are called **parallel programs.**
- The process of writing parallel programs is often referred to as **parallel programming.**

Parallel Computing



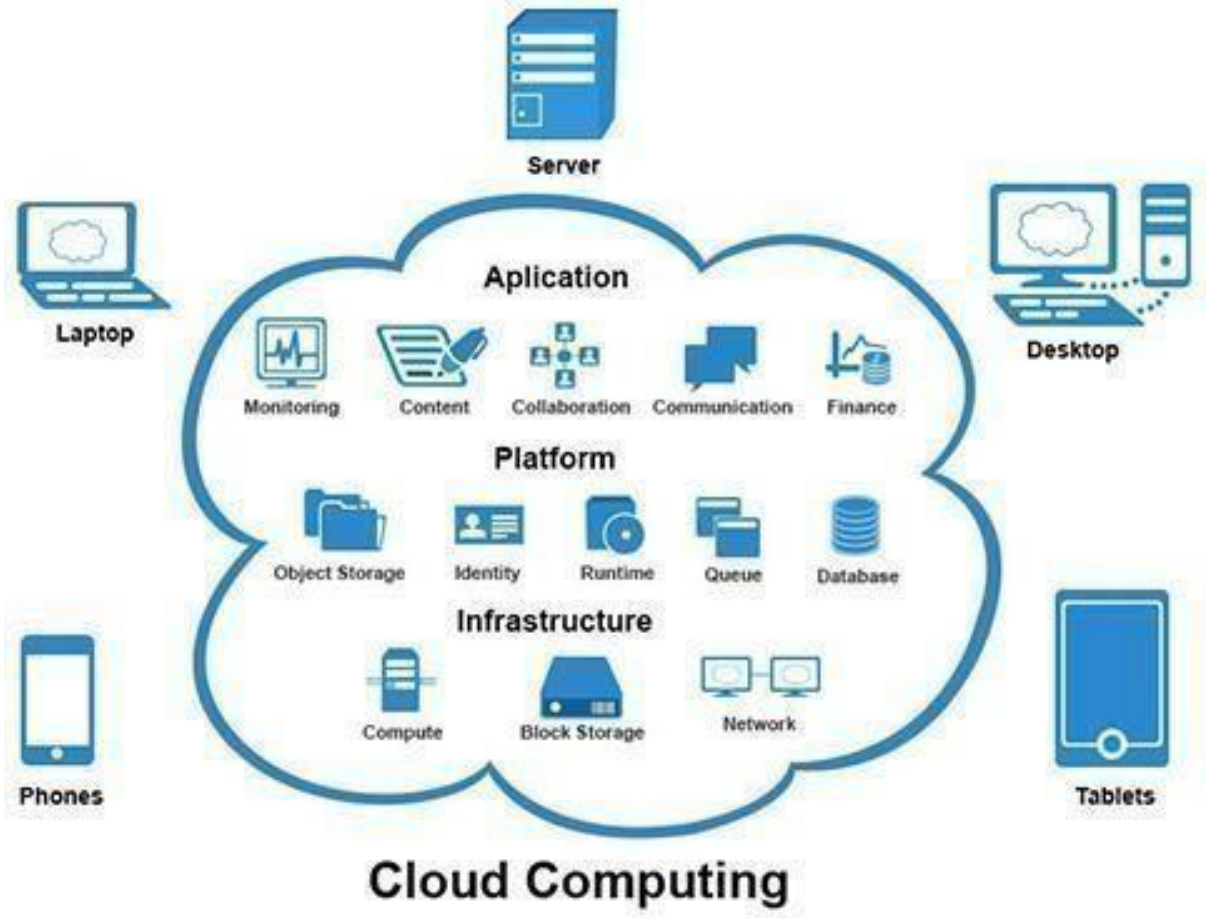
Distributed computing :

- A distributed system consists of multiple **autonomous computers**, each having its own private memory, communicating through a computer network.
- Information exchange in a distributed system is accomplished through message passing.
- A computer program that runs in a distributed system is known as a **distributed program.**
- The process of writing distributed programs is referred to as **distributed programming.**



Cloud computing :

- An Internet cloud of resources can be either a centralized or a distributed computing system.
- The cloud applies parallel or distributed computing, or both.
- Clouds can be built with physical or virtualized resources over large data centers that are centralized or distributed.



Grid Computing: Grid computing is the use of widely distributed computer resources to reach a common goal. A computing grid can be thought of as a distributed system with non-interactive workloads that involve many files. Grid computing is distinguished from conventional high-performance computing systems such as cluster computing in that grid computers have each node set to perform a different task/application. Grid computers also tend to be more heterogeneous and geographically dispersed (thus not physically coupled) than cluster computers.[1] Although a single grid can be dedicated to a particular application, commonly a grid is used for a variety of purposes. Grids are often constructed with general-purpose grid middleware software libraries.

UNIT- II

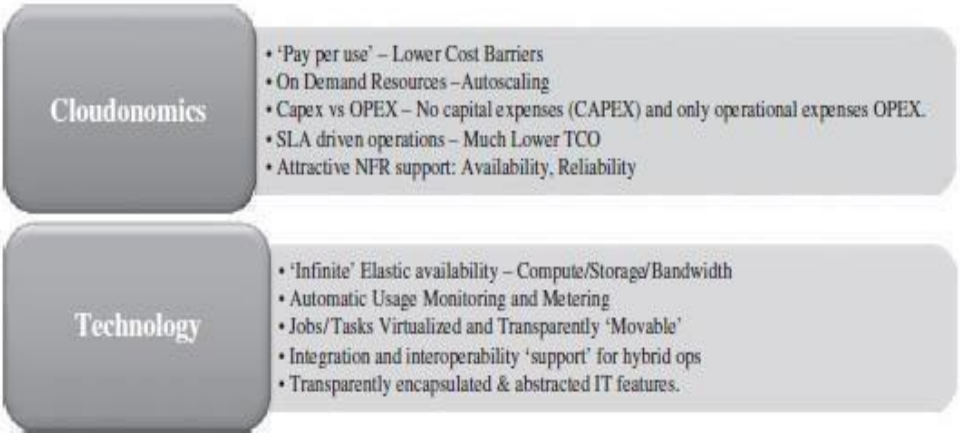
Migrating into a Cloud: Introduction, Broad Approaches to Migrating into the Cloud, the Seven-Step Model of Migration into a Cloud, Enriching the 'Integration as a Service' Paradigm for the Cloud Era, the Onset of Knowledge Era the Evolution of SaaS, Evolution of Saas.

Migrating into a Cloud

Cloud computing: "It is a techno-business disruptive model of using distributed large-scale data centers either private or public or hybrid offering customers a scalable virtualized infrastructure or an abstracted set of services qualified by service-level agreements (SLAs) and charged only by the abstracted IT resources consumed." Most enterprises today are powered by captive data centers. In most large or small enterprises today, IT is the backbone of their operations. Invariably for these large enterprises, their data centers are distributed across various geographies. They comprise systems and software that span several generations of products sold by a variety of IT vendors. In order to meet varying loads, most of these data centers are provisioned with capacity beyond the peak loads experienced. If the enterprise is in a seasonal or cyclical business, then the load variation would be significant. Thus what is observed generally is that the provisioned capacity of IT resources is several times the average demand. This is indicative of significant degree of idle capacity. Many data center management teams have been continuously innovating their management practices and technologies deployed to possibly squeeze out the last possible usable computing resource cycle through appropriate programming, systems configurations, SLAs, and systems management. Cloud computing turned attractive to them because they could pass on the additional demand from their IT setups onto the cloud while paying only for the usage and being unencumbered by the load of operations and management

The promise of the cloud computing services

In small and medium enterprises, cloud computing usage for all additional cyclical IT needs has yielded substantial and significant economic savings. This economics and the associated trade- offs, of leveraging the cloud computing services, now popularly called "cloudonomics," for satisfying enterprise's seasonal IT loads has become a topic of deep interest amongst IT managers and technology architects



As shown in Figure 2.1, the promise of the cloud both on the business front (the attractive cloudonomics) and the technology front widely aided the CxOs to spawnout several non-mission critical IT needs from the ambit of their captive traditional data centers to the appropriate cloud service. Invariably, these IT needs had some common features: They were typically Web- oriented; they represented seasonal IT demands; they were amenable to parallel batch processing; they were non-mission critical and therefore did not have high security demands. They included scientific applications too. Several small and medium business enterprises, however, leveraged the cloud much beyond the cautious user. Many startups opened their IT departments exclusively using cloud services—very successfully and with high ROI. Having observed these successes, several large enterprises have started successfully running pilots for leveraging the cloud. Many large enterprises run SAP to manage their operations.

Why Migrate

- Business Reasons
- Technological Reasons

What can be Migrated

- Application
- Code
- Design
- Architecture
- Usage

The migration of an enterprise application is best captured by the following:

$P \rightarrow P'_C + P'_l \square P'_{OFC} + P'_l$

where P is the application before migration, running in captive data center P'_C is the application part after migration into a (hybrid) cloud

P'_l is the part of application being run in the captive local data center P'_{OFC} is the application part optimized for cloud.

Invariably, migrating into the cloud is driven by economic reasons of cost cutting in both the IT capital expenses (Capex) as well as operational expenses(Opex).

If the average costs of using an enterprise application on a cloud is substantially lower than the costs of using it in one's captive data center and if the cost of migration does not add to the burden on ROI, then the case for migration into the cloud is strong.

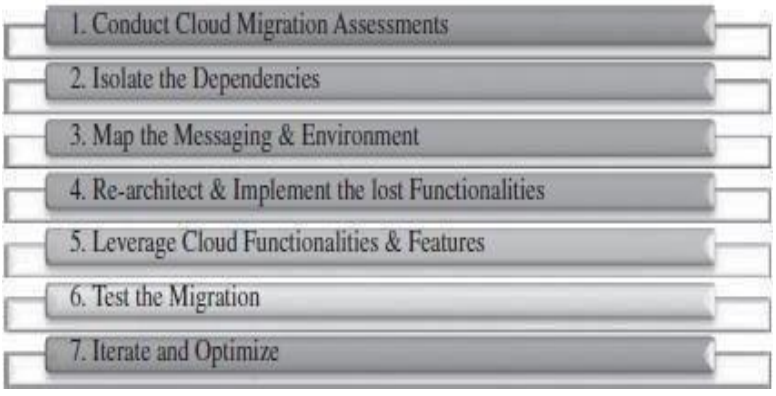
Apart from these costs, other factors that play a major role in the cloudonomics ofmigration are

- the licensing issues (for perhaps parts of the enterprise application)
- the SLA compliances, and the pricing of the cloud service offerings.

THE SEVEN-STEP MODEL OF MIGRATION INTO A CLOUD

Migration initiatives into the cloud are implemented in phases or in stages

- **Assessment**: Proof of concepts or prototypes for various approaches to the migration along with the leveraging of pricing parameters enables one to make appropriate assessments. These assessments are about the cost of migration as well as about the ROI that can be achieved in the case of production version
- **Isolation**: The next process step is in isolating all systemic and environmental dependencies of the enterprise application components within the captive data center. This, in turn, yields a picture of the level of complexity of the migration
- **Mapping**: Generating the mapping constructs between what shall possibly remainin the local captive data center and what goes onto the cloud.
- **Re-architect**: A substantial part of the enterprise application needs to be rearchitected, redesigned, and reimplemented on thecloud
- **Augment**: leverage the intrinsic features of the cloud computing service to augment our enterprise application in its own small ways
- **Test**: validate and test the new form of the enterprise application with an extensive test suite that comprises testing the components of the enterprise application on thecloud as well
- **optimize**: These test results could be positive or mixed. In the latter case, we iterate and optimize as appropriate. After several such optimizing iterations, the migration is deemed successful.



Unit -2 part 2

Virtualization:

Virtualization technology benefits the computer and IT industries by enabling users to share expensive hardware resources by multiplexing VMs on the same set of hardware hosts.

IMPLEMENTATION LEVELS OF VIRTUALIZATION

Virtualization is a computer architecture technology by which multiple virtual machines (VMs) are multiplexed in the same hardware machine. The idea of VMs can be dated back to the 1960s [53]. The purpose of a VM is to enhance resource sharing by many users and improve computer performance in terms of resource utilization and application flexibility. Hardware resources (CPU, memory, I/O devices, etc.) or software resources (operating system and software libraries) can be virtualized in various functional layers

The idea is to separate the hardware from the software to yield better system efficiency. For example, computer users gained access to much enlarged memory space when the concept of virtual memory was introduced

Levels of Virtualization Implementation A traditional computer runs with a host operating system specially tailored for its hardware architecture, as shown in Figure 3.1(a). After virtualization, different user applications managed by their own operating systems (guest OS) can run on the same hardware, independent of the host OS. This is often done by adding additional software, called a virtualization layer as shown in Figure 3.1(b). This virtualization layer is known as hypervisor or virtual machine monitor (VMM) [54]. The VMs are shown in the upper boxes, where applications run with their own guest OS over the virtualized CPU, memory, and I/O resources. The main function of the software layer for virtualization is to virtualize the physical hardware of a host machine into virtual resources to be used bythe VMs, exclusively

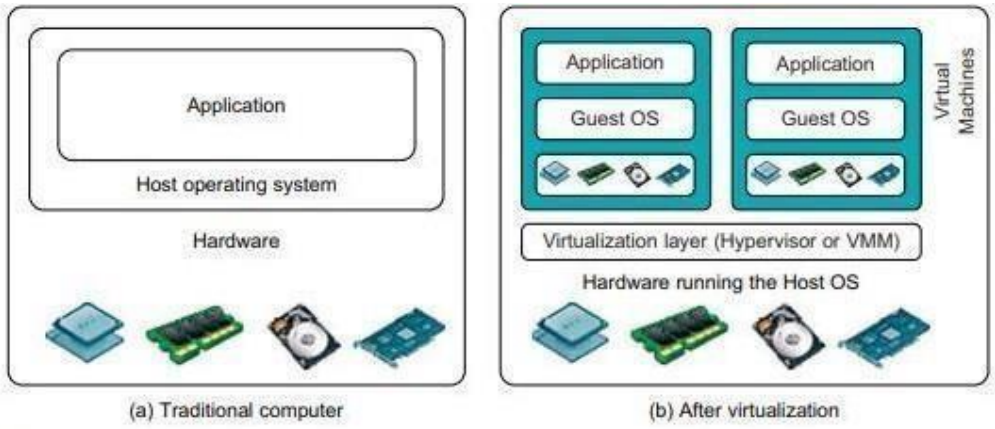
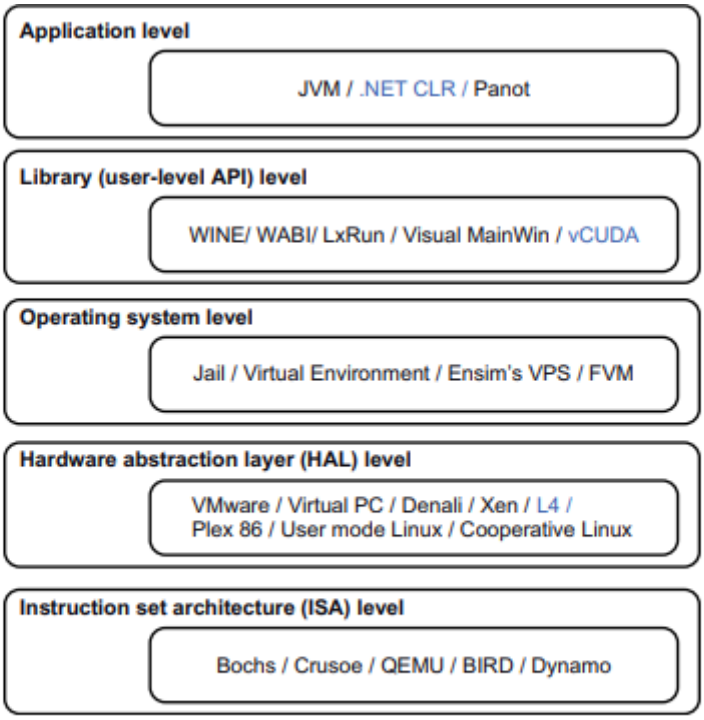


FIGURE 3.1
The architecture of a computer system before and after virtualization, where VMM stands for virtual machine monitor.



IGURE 3.2

/irtualization ranging from hardware to applications in five abstraction levels.

Instruction Set Architecture Level At the ISA level, virtualization is performed by emulatinga given ISA by the ISA of the host machine. For example, MIPS binary code can run on an x86-based host machine with the help of ISA emulation. With this approach, it is possible to run a large amount of legacy binary code written for various processors on any given new hardware host machine. Instruction set emulation leads to virtual ISAs created on any hardware machine. The basic emulation method is through code interpretation. An interpreter program interprets the source instructions to target instructions one by one. One source instruction may require tens or hundreds of native target instructions to perform its function. Obviously, this process is relatively slow. For better performance, dynamic binary translation is desired. This approach translates basic blocks of dynamic source instructions to target instructions. The basic blocks can also be extended to program traces or super blocks to increase translation efficiency. Instruction set emulation requires binary translation and optimization. A virtual instruction set architecture (V-ISA) thus requires adding a processor-specific software translation layer to the compiler.

Hardware Abstraction Level

Hardware Abstraction Level Hardware-level virtualization is performed right on top of the bare hardware. On the one hand, this approach generates a virtual hardware environment for a VM. On the other hand, the process manages the underlying hardware through virtualization. The idea is to virtualize a computer's resources, such as its processors, memory, and I/O devices. The intention is to upgrade the hardware utilization rate by multiple users concurrently.

Operating System Level This refers to an abstraction layer between traditional OS and user applications. OS-level virtualization creates isolated containers on a single physical server and the OS instances to utilize the hardware and software in data centers.

Library Support Level Most applications use APIs exported by user-level libraries rather than using lengthy system calls by the OS. Since most systems provide well-documented APIs, such an interface becomes another candidate for virtualization. Virtualization with library interfaces is possible by controlling the communication link between applications and the rest of a system through API hooks **User-Application Level Virtualization** at the application level virtualizes an application as a VM. On a traditional OS, an application often runs as a process. Therefore, application-level virtualization is also known as 132 CHAPTER 3 Virtual Machines and Virtualization of Clusters and Data Centers process-level virtualization. The most popular approach is to deploy high level language (HLL) VMs. In this scenario, the virtualization layer sits as an application program on top of the operating system, and the layer exports an abstraction of a VM that can run programs written and compiled to a particular abstract machine definition. Any program written in the HLL and compiled for this VM will be able to run on it. The Microsoft .NET CLR and Java Virtual Machine (JVM) are two good examples of this class of VM

Relative Merits of Different Approaches Table 3.1 compares the relative merits of implementing virtualization at various levels. The column headings correspond to four technical merits. "Higher Performance" and "Application Flexibility" are self-explanatory. "Implementation Complexity" implies the cost to implement that particular virtualization level. "Application Isolation" refers to the effort required to isolate resources committed to different VMs. Each row corresponds to a particular level of virtualization.

Table 3.1 Relative Merits of Virtualization at Various Levels (More "X"'s Means Higher Merit, with a Maximum of 5 X's)

Level of Implementation	Higher Performance	Application Flexibility	Implementation Complexity	Application Isolation
ISA	X	XXXXX	XXX	XXX
Hardware-level virtualization	XXXXX	XXX	XXXXX	XXXX
OS-level virtualization	XXXXX	XX	XXX	XX
Runtime library support	XXX	XX	XX	XX
User application level	XX	XX	XXXXX	XXXXX

There are three requirements for a VMM. First, a VMM should provide an environment for programs which is essentially identical to the original machine. Second, programs run in this environment should show, at worst, only minor decreases in speed. Third, a VMM should be in complete control of the system resources. Any program run under a VMM should exhibit a function identical to that which it runs on the original machine directly.

Table 3.2 Comparison of Four VMM and Hypervisor Software Packages

Provider and References	Host CPU	Host OS	Guest OS	Architecture
VMware Workstation [71]	x86, x86-64	Windows, Linux	Windows, Linux, Solaris, FreeBSD, Netware, OS/2, SCO, BeOS, Darwin	Full Virtualization
VMware ESX Server [71]	x86, x86-64	No host OS	The same as VMware Workstation	Para-Virtualization
Xen [7,13,42]	x86, x86-64, IA-64	NetBSD, Linux, Solaris	FreeBSD, NetBSD, Linux, Solaris, Windows XP and 2003 Server	Hypervisor
KVM [31]	x86, x86-64, IA-64, S390, PowerPC	Linux	Linux, Windows, FreeBSD, Solaris	Para-Virtualization

Virtualization Support at the OS Level

With the help of VM technology, a new computing mode known as cloud computing is emerging.. However, cloud computing has at least two challenges.

- The first is the ability to use a variable number of physical machines and VM instances depending on the needs of a problem. For example, a task may need only a single CPU during some phases of execution but may need hundreds of CPUs at other times.
- The second challenge concerns the slow operation of instantiating new VMs. Currently, new VMs originate either as fresh boots or as replicates of a template VM, unaware of the current application state.

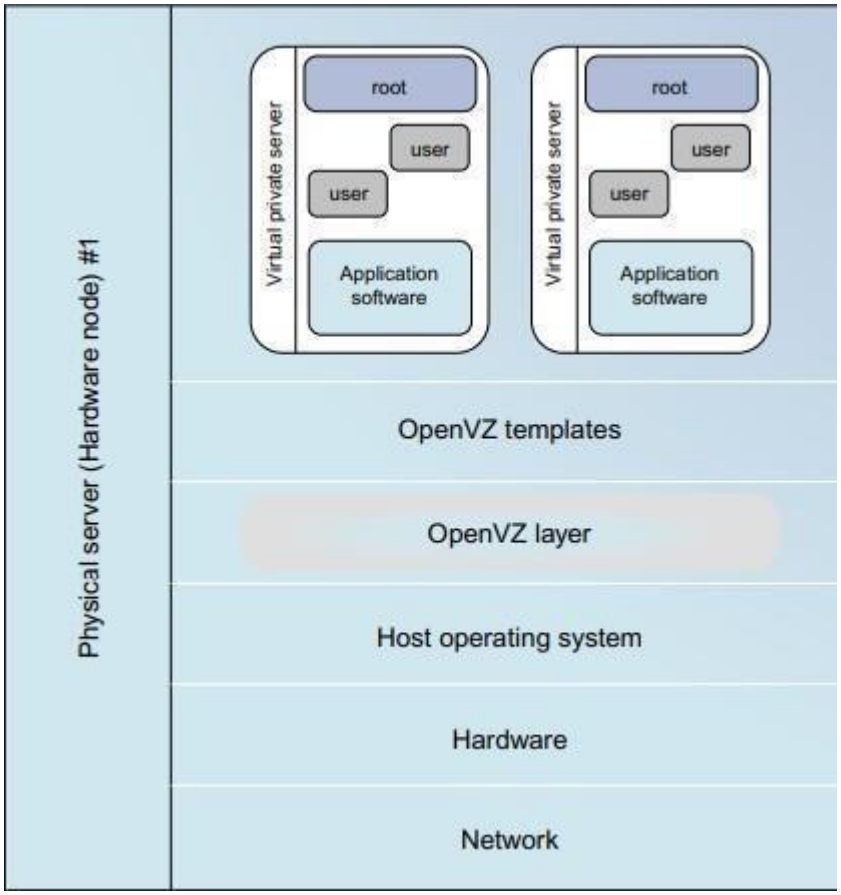
Why OS Level Virtualization ?

OS-level virtualization provides a feasible solution for these hardware-level virtualization issues. Operating system virtualization inserts a virtualization layer inside an operating system to partition a machine's physical resources. It enables multiple isolated VMs within a single operating system kernel. This kind of VM is often called a virtual execution environment (VE), Virtual Private System (VPS), or simply container. From the user's point of view, VEs look like real servers. This means a VE has it s own set of processes, file system, user accounts, network interfaces with IP addresses, routing tables, firewall rules, and other personal settings.

2 Advantages of OS Extensions

Compared to hardware-level virtualization, the benefits of OS extensions are twofold: (1) VMs at the operating system level have minimal startup/shutdown costs, low resource requirements, and high scalability; and (2) for an OS-level VM, it is possible for a VM and its host environment to synchronize state changes when necessary. These benefits can be achieved via two mechanisms of OS-level virtualization: (1) All OS-level VMs on the same physical machine share a single operating system kernel; and (2) the virtualization layer can be designed in a way that allows processes in VMs to access as many resources of the host machine as possible, but never to modify them. In cloud 3.1

Implementation Levels of Virtualization 135 computing, the first and second benefits can be used to overcome the defects of slow initialization of VMs at the hardware level, and being unaware of the current application state, respectively



3 Disadvantages of OS Extensions The main disadvantage of OS extensions is that all the VMs at operating system level on a single container must have the same kind of guest operating system. That is, although different OS-level VMs may have different operating system distributions, they must pertain to the same operating system family.

VIRTUALIZATION STRUCTURES/TOOLS AND MECHANISMS

The virtualization layer is responsible for converting portions of the real hardware into virtual hardware. Therefore, different operating systems such as Linux and Windows can run on the same physical machine, simultaneously. Depending on the position of the virtualization layer, there are several classes of VM architectures, namely the hypervisor architecture, paravirtualization, and host-based virtualization. The hypervisor is also known as the VMM (Virtual Machine Monitor)

Hypervisor and Xen Architecture

The hypervisor supports hardware-level virtualization (see Figure 3.1(b)) on bare metal devices like CPU, memory, disk and network interfaces. The hypervisor software sits directly between the physical hardware and its OS. This virtualization layer is referred to as either the VMM or the hypervisor. The hypervisor provides hypercalls for the guest OSes and applications. Depending on the functionality, a hypervisor can assume a micro-kernel architecture like the Microsoft Hyper-V. Or it can assume a monolithic hypervisor architecture like the VMware ESX for server virtualization. A micro-kernel hypervisor includes only the basic and unchanging functions (such as physical memory management and processor scheduling). The device drivers and other changeable components are outside the hypervisor. A monolithic hypervisor implements all the aforementioned functions, including those of the device drivers. Therefore, the size of the hypervisor code of a micro-kernel hypervisor is smaller than that of a monolithic hypervisor. Essentially, a hypervisor must be able to convert physical devices into virtual resources dedicated for the deployed VM to use.

The Xen Architecture Xen is an open source hypervisor program developed by Cambridge University. Xen is a microkernel hypervisor, which separates the policy from the mechanism. The Xen hypervisor implements all the mechanisms, leaving the policy to be handled by Domain 0, as shown in Figure 3.5. Xen does not include any device drivers natively [7]. It just provides a mechanism by which a guest OS can have direct access to the physical devices. As a result, the size of the Xen hypervisor is kept rather small. Xen provides a virtual environment located between the hardware and the OS. A number of vendors are in the process of developing commercial Xen hypervisors, among them are Citrix XenServer [62] and Oracle VM [42]. The core components of a Xen system are the hypervisor, kernel, and applications. The organization of the three components is important. Like other virtualization systems, many guest OSes can run on top of the hypervisor. However, not all guest OSes are created equal, and one in

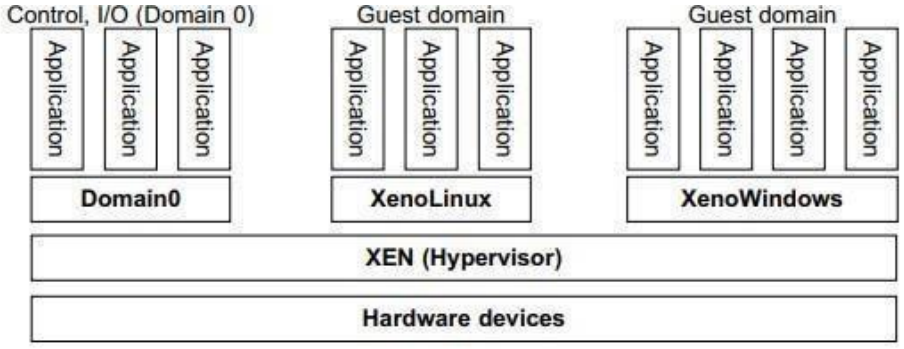


FIGURE 3.5

The Xen architecture's special domain 0 for control and I/O, and several guest domains for user applications.

Binary Translation with Full Virtualization

Depending on implementation technologies, hardware virtualization can be classified into two categories: **full virtualization** and **host-based virtualization**. Full virtualization does not need to modify the host OS. It relies on binary translation to trap and to virtualize the execution of certain sensitive, nonvirtualizable instructions.

With full virtualization, noncritical instructions run on the hardware directly while critical instructions are discovered and replaced with traps into the VMM to be emulated by software

Host-Based Virtualization An alternative VM architecture is to install a virtualization layer on top of the host OS. This host OS is still responsible for managing the hardware. The guest OSes are installed and run on top of the virtualization layer. Dedicated applications may run on the VMs. Certainly, some other applications can also run with the host OS directly. This hostbased architecture has some distinct advantages, as enumerated next. First, the user can install this VM architecture without modifying the host OS. The virtualizing software can rely on the host OS to provide device drivers and other low-level services. This will simplify the VM design and ease its deployment. Second, the host-based approach appeals to many host machine configurations. Compared to the hypervisor/VMM architecture, the performance of the host-based architecture may also be low. When an application requests hardware access, it involves four layers of mapping which downgrades performance significantly. When the ISA of a guest OS is different from the ISA of

Para-Virtualization with Compiler Support Para-virtualization needs to modify the guest operating systems. A para-virtualized VM provides special APIs requiring substantial OS modifications in user applications. Performance degradation is a critical issue of a virtualized system. No one wants to use a VM if it is much slower than using a physical machine. The virtualization layer can be inserted at different positions in a machine software stack. However, para-virtualization attempts to reduce the virtualization overhead, and thus improve performance by modifying only the guest OS kernel. Figure 3.7 illustrates the concept of a para-virtualized VM architecture. The guest operating systems are para-virtualized. They are assisted by an intelligent compiler to replace the nonvirtualizable OS instructions by hypercalls as illustrated in Figure 3.8. The traditional x86 processor offers four instruction execution rings: Rings 0, 1, 2, and 3. The lower the ring number, the higher the privilege of instruction being executed. The OS is responsible for managing the hardware and the privileged instructions to execute at Ring 0, while user-level applications run at Ring 3. The best example of para-virtualization is the KVM to be described below.

3.2.3.1 Para-Virtualization Architecture When the x86 processor is virtualized, a virtualization layer is inserted between the hardware and the OS. According to the x86 ring definition, the virtualization layer should also be installed at Ring 0. Different instructions at Ring 0 may cause some problems. In Figure 3.8, we show that para-virtualization replaces nonvirtualizable instructions with hyper calls that communicate directly with the hypervisor or VMM. However, when the guest OS kernel is modified for virtualization, it can no longer run on the hardware directly.

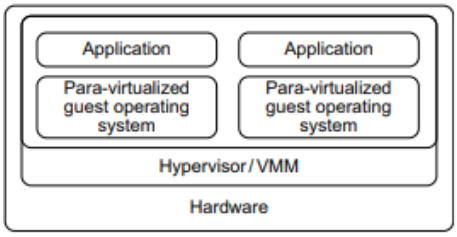


FIGURE 3.7
Para-virtualized VM architecture, which involves modifying the guest OS kernel to replace nonvirtualizable instructions with hypercalls for the hypervisor or the VMM to carry out the virtualization process (See Figure 3.8 for more details.)

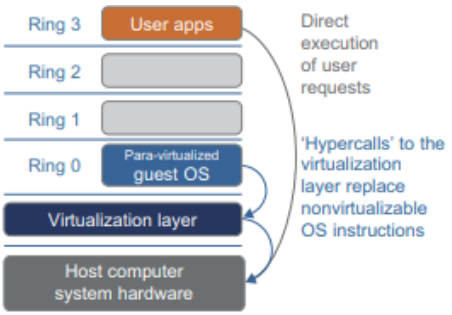


FIGURE 3.8
The use of a para-virtualized guest OS assisted by an intelligent compiler to replace nonvirtualizable OS instructions by hypercalls.
(Courtesy of VMWare [71])

Para-Virtualization with Compiler Support

Unlike the full virtualization architecture which intercepts and emulates privileged and sensitive instructions at runtime, para-virtualization handles these instructions at compile time. The guest OS kernel is modified to replace the privileged and sensitive instructions with hypercalls to the hypervisor or VMM. Xen assumes such a para-virtualization architecture. The guest OS running in a guest domain may run at Ring 1 instead of at Ring 0. This implies that the guest OS may not be able to execute some privileged and sensitive instructions. The privileged instructions are implemented by hypercalls to the hypervisor. After replacing the instructions with hypercalls, the modified guest OS emulates the behavior of the original guest OS

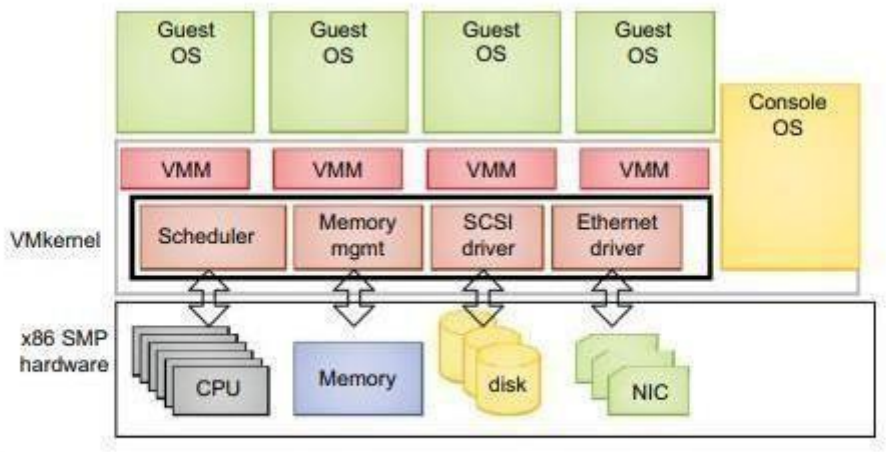


FIGURE 3.9
The VMware ESX server architecture using para-virtualization.

VIRTUALIZATION OF CPU, MEMORY, AND I/O DEVICES

To support virtualization, processors such as the x86 employ a special running mode and instructions, known as hardware-assisted virtualization. In this way, the VMM and guest OS run in different modes and all sensitive instructions of the guest OS and its applications are trapped in the VMM.

Hardware Support for Virtualization

Modern operating systems and processors permit multiple processes to run simultaneously. If there is no protection mechanism in a processor, all instructions from different processes will access the hardware directly and cause a system crash. Therefore, all processors have at least two modes, user mode and supervisor mode, to ensure controlled access of critical hardware. Instructions running in supervisor mode are called privileged instructions. Other instructions are unprivileged instructions

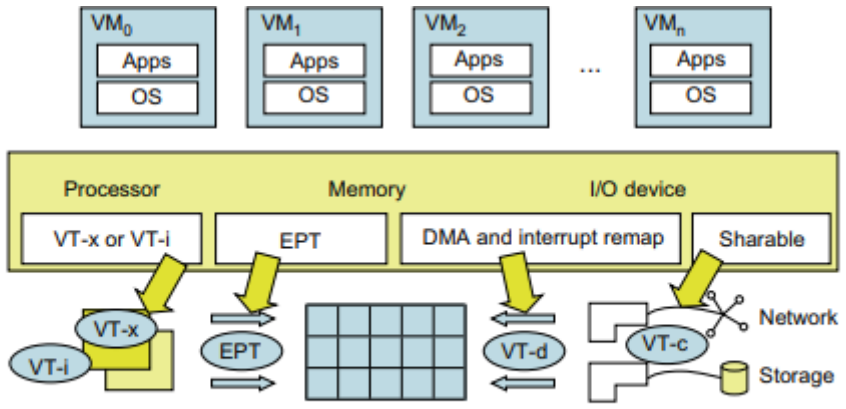


FIGURE 3.10
Intel hardware support for virtualization of processor, memory, and I/O devices.

CPU Virtualization A VM is a duplicate of an existing computer system in which a majority of the VM instructions are executed on the host processor in native mode. Thus, unprivileged instructions of VMs run directly on the host machine for higher efficiency. Other critical instructions should be handled carefully for correctness and stability. The critical instructions are divided into three categories: privileged instructions, controlsensitive instructions, and behavior-sensitive instructions. Privileged instructions execute in a privileged mode and will be trapped if executed outside this mode. Control-sensitive instructions attempt to change the configuration of resources used. Behavior-sensitive instructions have different behaviors depending on the configuration of resources, including the load and store operations over the virtual memory. A CPU architecture is virtualizable if it supports the ability to run the VM's privileged and unprivileged instructions in the CPU's user mode while the VMM runs in supervisor mode. When the privileged instructions including control- and behavior-sensitive instructions of a VM are executed, they are trapped in the VMM. In this case, the VMM acts as a unified mediator for hardware access from different VMs to guarantee the correctness and stability of the whole system. However, not all CPU architectures are virtualizable. RISC CPU architectures can be naturally virtualized because all control- and behavior-sensitive instructions are privileged instructions. On the contrary, x86 CPU architectures are not primarily designed to support virtualization. This is because about 10 sensitive instructions, such as SGDT and SMSW, are not privileged instructions. When these instructions execute in virtualization they cannot be trapped in the VMM.

Hardware-Assisted CPU Virtualization This technique attempts to simplify virtualization because full or para virtualization is complicated. Intel and AMD add an additional mode called privilege mode level (some people call it Ring-1) to x86 processors. Therefore, operating systems can still run at Ring 0 and the hypervisor can run at Ring -1. All the privileged and sensitive instructions are trapped in the hypervisor automatically. This technique removes the difficulty of implementing binary translation of full virtualization. It also lets the operating system run in VMs without modification.

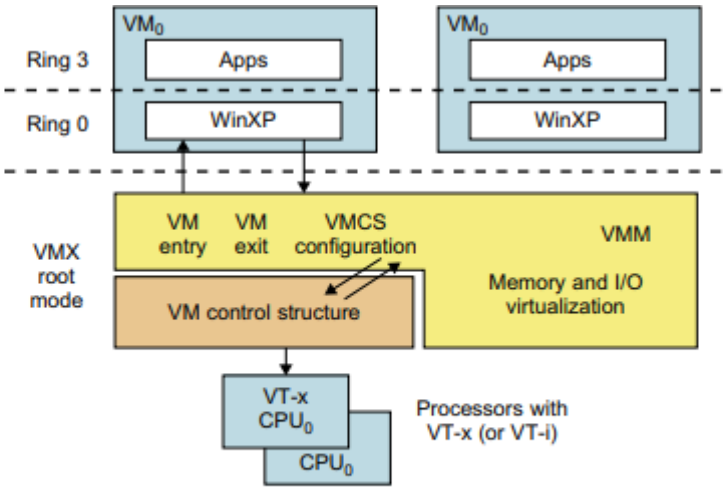


FIGURE 3.11
Intel hardware-assisted CPU virtualization.

Memory Virtualization Virtual memory virtualization is similar to the virtual memory support provided by modern operating systems. In a traditional execution environment, the operating system maintains mappings of virtual memory to machine memory using page tables, which is a one-stage mapping from virtual memory to machine memory. All modern x86 CPUs include a memory management unit (MMU) and a translation lookaside buffer (TLB) to optimize virtual memory performance. However, in a virtual execution environment, virtual memory virtualization involves sharing the physical system memory in RAM and dynamically allocating it to the physical memory of the VMs. That means a two-stage mapping process should be maintained by the guest OS and the VMM, respectively: virtual memory to physical memory and physical memory to machine memory. Furthermore, MMU virtualization should be supported, which is transparent to the guest OS. The guest OS continues to control the mapping of virtual addresses to the physical memory addresses of VM

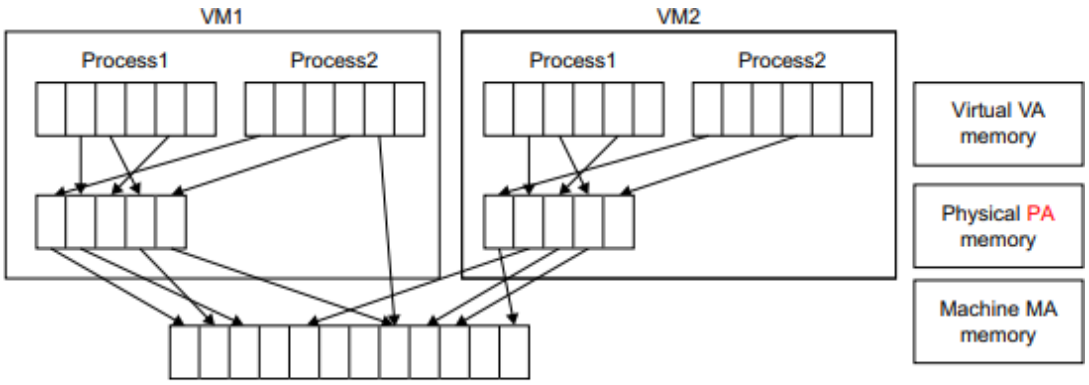


FIGURE 3.12
Two-level memory mapping procedure.

I/O Virtualization I/O virtualization involves managing the routing of I/O requests between virtual devices and the shared physical hardware. At the time of this writing, there are three ways to implement I/O virtualization: full device emulation, para-virtualization, and direct I/O. Full device emulation is the first approach for I/O virtualization. Generally, this approach emulates well-known, real-world devices

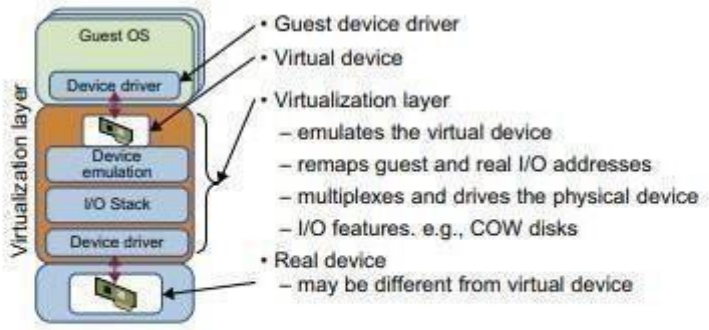


FIGURE 3.14
Device emulation for I/O virtualization implemented inside the middle layer that maps real I/O devices int virtual devices for the guest device driver to use.

Virtualization in Multi-Core Processors Virtualizing a multi-core processor is relatively more complicated than virtualizing a uni-core processor. Though multicore processors are claimed to have higher performance by integrating multiple processor cores in a single chip, muti-core virtualiuzation has raised some new challenges to computer architects, compiler constructors, system designers, and application programmers. There are mainly two difficulties: Application programs must be parallelized to use all cores fully, and software must explicitly assign tasks to the cores, which is a very complex problem.

Physical versus Virtual Processor Cores Wells, et al. [74] proposed a multicore virtualization method to allow hardware designers to get an abstraction of the low-level details of the processor cores. This technique alleviates the burden and inefficiency of managing hardware resources by software. It is located under the ISA and remains unmodified by the operating system or VMM (hypervisor). Figure 3.16 illustrates the technique of a software-visible VCPU moving from one core to another and temporarily suspending execution of a VCPU when there are no appropriate cores on which it can run.

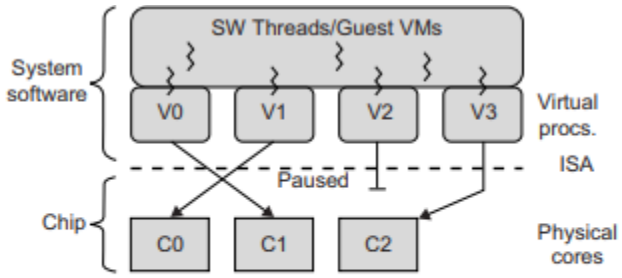
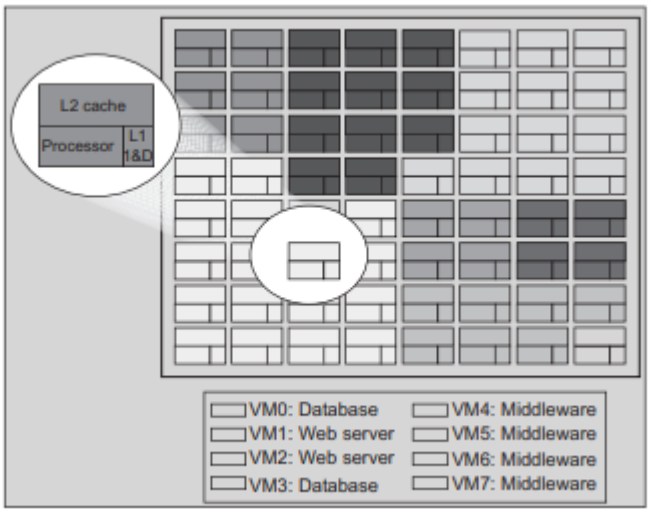


FIGURE 3.16

Multicore virtualization method that exposes four VCPUs to the software, when only three cores are actually present.

Virtual Hierarchy The emerging many-core chip multiprocessors (CMPs) provides a new computing landscape. Instead of supporting time-sharing jobs on one or a few cores, we can use the abundant cores in a space-sharing, where single-threaded or multithreaded jobs are simultaneously assigned to separate groups of cores for long time intervals. This idea was originally suggested by Marty and Hill [39]. To optimize for space-shared workloads, they propose using virtual hierarchies to overlay a coherence and caching hierarchyonto a physical processor

Today's many-core CMPs use a physical hierarchy of two or more cache levels that statically determine the cache allocation and mapping. A virtual hierarchy is a cache hierarchy that can adapt to fit the workload or mix of workloads [39]. The hierarchy's first level locates data blocks close to the cores needing them for faster access, establishes a shared-cache domain, and establishes a point of coherence for faster communication. When a miss leaves a tile, it first attempts to locate the block (or sharers) within the first level. The first level can also provide isolation between independent workloads.



(a) Mapping of VMs into adjacent cores

(b) Multiple virtual clusters assigned to various workloads

FIGURE 3.17

CMP server consolidation by space-sharing of VMs into many cores forming multiple virtual clusters to execute various workloads.

VIRTUAL CLUSTERS AND RESOURCE MANAGEMENT

A physical cluster is a collection of servers (physical machines) interconnected by a physical network such as a LAN.

1 Physical versus Virtual Clusters Virtual clusters are built with VMs installed at distributed servers from one or more physical clusters. The VMs in a virtual cluster are interconnected logically by a virtual network across several physical networks. Figure 3.18 illustrates the concepts of virtual clusters and physical clusters. Each virtual cluster is formed with physical machines or a VM hosted by multiple physical clusters. The virtual cluster boundaries are shown as distinct boundaries.

The provisioning of VMs to a virtual cluster is done dynamically to have the following interesting properties:

- The virtual cluster nodes can be either physical or virtual machines. Multiple VMs running with different OSes can be deployed on the same physical node.
- A VM runs with a guest OS, which is often different from the host OS, that manages the resources in the physical machine, where the VM is implemented.
- The purpose of using VMs is to consolidate multiple functionalities on the same server. This will greatly enhance server utilization and application flexibility

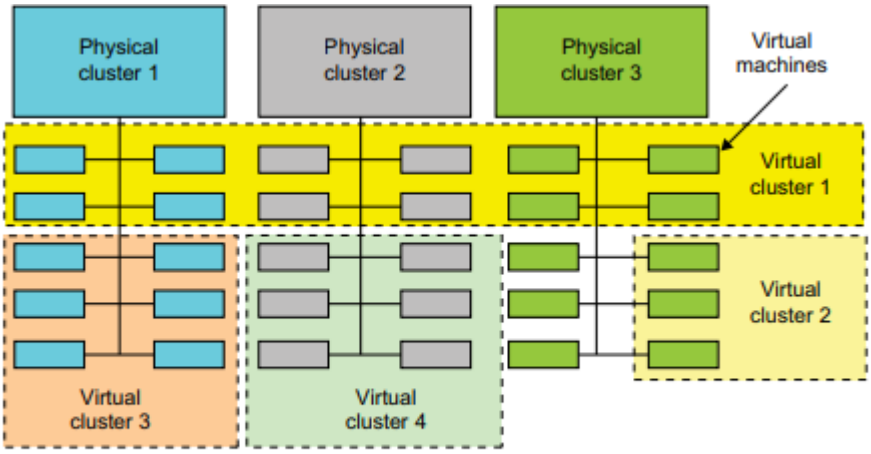


FIGURE 3.18

A cloud platform with four virtual clusters over three physical clusters shaded differently.

VMs can be colonized (replicated) in multiple servers for the purpose of promoting distributed parallelism, fault tolerance, and disaster recovery.

• The size (number of nodes) of a virtual cluster can grow or shrink dynamically, similar to the way an overlay network varies in size in a peer-to-peer (P2P) network.

• The failure of any physical nodes may disable some VMs installed on the failing nodes. But the failure of VMs will not pull down the host system.

VIRTUALIZATION FOR DATA-CENTER Data centers have grown rapidly in recent years, and all major IT companies are pouring their resources into building new data centers. In addition, Google, Yahoo!, Amazon, Microsoft, HP, Apple, and IBM are all in the game. All these companies have invested billions of dollars in datacenter construction and automation. Data-center automation means that huge volumes of hardware, software, and database resources in these data centers can be allocated dynamically to millions of Internet users simultaneously, with guaranteed QoS and cost-effectiveness.

The latest virtualization development highlights high availability (HA), backup services, workload balancing, and further increases in client bases. IDC projected that automation, service orientation, policy-based, and variable costs in the virtualization market. The total business opportunities may increase to $3.2 billion by 2011. The major market share moves to the areas of HA, utility computing, production consolidation, and client bases. In what follows, we will discuss server consolidation, virtual storage, OS support, and trust management in automated data-center designs.

Server Consolidation in Data Centers In data centers, a large number of heterogeneous workloads can run on servers at various times. These heterogeneous workloads can be roughly divided into two categories: chatty workloads and non interactive workloads. Chatty workloads may burst at some point and return to a silent state at some other point. A web video service is an example of this, whereby a lot of people use it at night and few people use it during the day. Non interactive workloads do not require

people's efforts to make progress after they are submitted. High-performance computing is a typical example of this. At various stages, the requirements for resources of these workloads are dramatically different. However, to guarantee that a workload will always be able to cope with all demand levels, the workload is statically allocated enough resources so that peak demand is satisfied. Figure 3.29 illustrates server virtualization in a data center

In general, the use of VMs increases resource management complexity. This causes a challenge in terms of how to improve resource utilization as well as guarantee QoS in data centers. In detail, server virtualization has the following side effects:

• Consolidation enhances hardware utilization. Many underutilized servers are consolidated into fewer servers to enhance resource utilization. Consolidation also facilitates backup services and disaster recovery.

• This approach enables more agile provisioning and deployment of resources. In a virtual environment, the images of the guest OSes and their applications are readily cloned and reused.

• The total cost of ownership is reduced. In this sense, server virtualization causes deferred purchases of new servers, a smaller data-center footprint, lower maintenance costs, and lower power, cooling, and cabling requirements.

• This approach improves availability and business continuity. The crash of a guest OS has no effect on the host OS or any other guest OS. It becomes easier to transfer a VM from one server to another, because virtual servers are unaware of the underlying hardware.

Cloud OS for Virtualized Data Centers

Data centers must be virtualized to serve as cloud providers. Table 3.6 summarizes four virtual infrastructure (VI) managers and OSes. These VI managers and OSes are specially tailored for virtualizing data centers which often own a large number of servers in clusters. Nimbus, Eucalyptus, and OpenNebula are all open source software available to the general public. Only vSphere 4 is a proprietary OS for cloud resource virtualization and management over data centers

Table 3.6 VI Managers and Operating Systems for Virtualizing Data Centers [9]

Manager/ OS, Platforms, License	Resources Being Virtualized, Web Link	Client API, Language	Hypervisors Used	Public Cloud Interface	Special Features
Nimbus Linux, Apache v2	VM creation, virtual cluster, www .nimbusproject.org/	EC2 WS, WSRF, CLI	Xen, KVM	EC2	Virtual networks
Eucalyptus Linux, BSD	Virtual networking (Example 3.12 and [41]), www .eucalyptus.com/	EC2 WS, CLI	Xen, KVM	EC2	Virtual networks
OpenNebula Linux, Apache v2	Management of VM, host, virtual network, and scheduling tools, www.opennebula.org/	XML-RPC, CLI, Java	Xen, KVM	EC2, Elastic Host	Virtual networks, dynamic provisioning
vSphere 4 Linux, Windows, proprietary	Virtualizing OS for data centers (Example 3.13), www .vmware.com/ products/vsphere/ [66]	CLI, GUI, Portal, WS	VMware ESX, ESXi	VMware vCloud partners	Data protection, vStorage, VMFS, DRM, HA

Eucalyptus is an open source software system (Figure 3.27) intended mainly for supporting Infrastructure as a Service (IaaS) clouds. The system primarily supports virtual networking and the management of VMs; virtual storage is not supported. Its purpose is to build private clouds that can interact with end users through Ethernet or the Internet. The system also supports interaction with other private clouds or public clouds over the Internet. The system is short on security and other desired features for general-purpose grid or cloud applications

UNIT- III

Infrastructure as a Service (IAAS) & Platform (PAAS): Virtual machines provisioning and Migration services, Virtual Machines Provisioning and Manageability, Virtual Machine Migration Services, VM Provisioning and Migration in Action. On the Management of Virtual machines for Cloud Infrastructures- Aneka—Integration of Private and Public Clouds.

INFRASTRUCTURE AS A SERVICE PROVIDERS

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

Features

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

Geographic Presence: To improve availability and responsiveness, a provider of worldwide services would typically build several data centers distributed around the world. For example, Amazon Web Services presents the concept of "availability zones" and "regions" for its EC2service. Availability zones are "distinct locations that are engineered to be insulated from failures in other availability zones and provide inexpensive, low-latency network connectivity to other availability zones in the same region." Regions, in turn, "are geographically dispersed and will be in separate geographic areas or countries

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

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

Automatic Scaling and Load Balancing: Elasticity is a key characteristic of the

cloud computing model. Applications often need to scale up and down to meet varying load conditions. Automatic scaling is a highly desirable feature of IaaS clouds. It allow users to set conditions for when they want their applications to scale up and down, based on application-specific metrics such as transactions per second, number of simultaneous users, request latency, and so forth. When the number of virtual servers is increased by automatic scaling, incoming traffic must be automatically distributed among the available servers. This activity enables applications to promptly respond to traffic increase while also achieving greater fault tolerance.

Service-Level Agreement: Service-level agreements (SLAs) are offered by IaaS providers to express their commitment to delivery of a certain QoS. To customers it serves as a warranty. An SLA usually include availability and performance guarantees. Additionally, metrics must be agreed upon by all parties as well as penalties for violating these expectations. Most IaaS providers focus their SLA terms on availability guarantees, specifying the minimum percentage of time the system will be available during a certainperiod.

Hypervisor and Operating System Choice: Traditionally, IaaS offerings have been based on heavily customized open-source Xen deployments. IaaS providers needed expertise in Linux, networking, virtualization, metering, resource management, and many other low-level aspects to successfully deploy and maintain their cloud offerings.

Case Studies

Amazon Web Services: Amazon WS4 (AWS) is one of the major players in the cloud computing market. It pioneered the introduction of IaaS clouds in 2006. It offers a variety cloud services, most notably: S3 (storage), EC2 (virtual servers), Cloudfront (content delivery), Cloudfront Streaming (video streaming), SimpleDB (structured datastore), RDS (Relational Database), SQS (reliable messaging), and Elastic MapReduce (data processing). The ElasticCompute Cloud (EC2) offers Xen-based virtual servers (instances) that can be instantiated from Amazon Machine Images (AMIs). Instances are available in a variety of sizes, operating systems, architectures, and price. CPU capacity of instances is measured in Amazon Compute Units and, although fixed for each instance, vary among instance types from 1 (small instance) to 20 (high CPU instance). Each instance provides a certain amount of nonpersistent disk space; a persistence disk service (Elastic Block Storage) allows attaching virtual disks to instances with space up to 1TB. Elasticity can be achieved by combining the CloudWatch, Auto Scaling,andElastic Load Balancing features, which allow the number of instances to scale up and down automatically based on a set of customizable rules, and traffic to be distributed across available instances. Fixed IP address (Elastic IPs) are not available by default, but can be obtained at an additional cost.

Flexiscale: Flexiscale is a UK-based provider offering services similar in nature to Amazon Web Services. Flexiscale cloud provides the following features: available in UK; Web services (SOAP), Web-based user interfaces; access to virtual server mainly via SSH (Linux) and Remote Desktop (Windows); 100% availability SLA with automatic recovery of VMs in case of hardware failure; per hour pricing;

Linux and Windows operating systems; automatic scaling (horizontal/vertical).

Joyent: Joyent's Public Cloud offers servers based on Solaris containers virtualization technology. These servers, dubbed accelerators, allow deploying various specialized software- stack based on a customized version of Open- Solaris operating system, which include by default a Web-based configuration tool and several pre-installed software, such as Apache, MySQL, PHP, Ruby on Rails, and Java. Software load balancing is available as an accelerator in addition to hardware load balancers. A notable feature of Joyent's virtual servers is automatic vertical scaling of CPU cores, which means a virtual server can make use of additional CPUs automatically up to the maximum number of cores available in the physical host.
The Joyent public cloud offers the following features: multiple geographic locations inthe United States; Web-based user interface; access to virtual server via SSH and Web-based administration tool; 100% availability SLA; per month pricing; OS-level virtualization Solaris containers; Open- Solaris operating systems; automatic scaling(vertical).

GoGrid: GoGrid, like many other IaaS providers, allows its customers to utilize a range of pre- made Windows and Linux images, in a range of fixed instance sizes. GoGrid also offers "value- added" stacks on top for applications such as high-volume Web serving, e-Commerce, and database stores. It offers some notable features, such as a "hybrid hosting" facility, which combines traditional dedicated hosts with auto-scaling cloud server infrastructure. As part of its core IaaS offerings, GoGrid also provides free hardware load balancing, auto-scaling capabilities, and persistent storage, features that typically add an additional cost for most other IaaS providers.

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

PLATFORM AS A SERVICE PROVIDERS

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

Features

Programming Models, Languages, and Frameworks: Programming models made available by IaaS providers define how users can express their applications using higher levels of abstraction and efficiently run them on the cloud platform. Each model aims at efficiently solving aparticular problem. In the cloud computing domain, the most common activities that require specialized models are: processing of large dataset in clusters of computers (MapReduce model),

development of request-based Web services and applications; definition and orchestration of business processes in the form of workflows (Workflow model); and high-performance distributed execution of various computational tasks.

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

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

Persistence Options: A persistence layer is essential to allow applications to record their state and recover it in case of crashes, as well as to store user data. Web and enterprise application developers have chosen relational databases as the preferred persistence method. These databases offer fast and reliable structured data storage and transaction processing, but may lack scalability to handle several petabytes of data stored in commodity computers. In the cloud computing domain, distributed storage technologies have emerged, which seek to be robust and highly scalable, at the expense of relational structure and convenient query languages.

Case Studies

Aneka: Aneka is a .NET-based service-oriented resource management and development platform. Each server in an Aneka deployment (dubbed Aneka cloud node) hosts the Aneka container, which provides the base infrastructure that consists of services for persistence, security (authorization, authentication and auditing), and communication (message handling and dispatching). Cloud nodes can be either physical server, virtual machines (XenServer and VMware are supported), and instances rented from Amazon EC2. The Aneka container can also host any number of optional services that can be added by developers to augment the capabilities of an Aneka Cloud node, thus providing a single, extensible framework for orchestrating various applicationmodels.

Several programming models are supported by such task models to enable execution of legacy HPC applications and MapReduce, which enables a variety of data-mining and search applications. Users request resources via a client to a reservation services manager of the Aneka master node, which manages all cloud nodes and contains scheduling service to distribute request to cloud nodes.

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

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

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

Public Cloud and Infrastructure Services

Public cloud or external cloud

- Resources are dynamically provisioned via publicly accessible Web applications/Web services (SOAP or RESTful interfaces) from an off-site third-party provider
- Shares resources and bills on a fine-grained utility computing basis
- The user pays only for the capacity of the provisioned resources at a particular time
- Examples for vendors who publicly provide IaaS
  - Amazon Elastic Compute Cloud (EC2), GoGrid, Joyent Accelerator, Rackspace, AppNexus, FlexiScale, and Manjrasoft Aneka

Amazon Elastic Compute Cloud (EC2) is an IaaS service

- Provides elastic compute capacity in the cloud
- Leveraged via Web services (SOAP or REST), a Web-based AWS (Amazon Web Service) management console, or the EC2 command line tools
- Provides hundreds of pre-made AMIs (Amazon Machine Images) with a variety of operating systems and pre-loaded software
  - i.e., Linux,OpenSolaris, or Windows
- Provides complete control of computing resources run on Amazon's computing and infrastructure environment easily
- Reduces the time required for obtaining and booting a new server's instances to minutes
- Allows a quick scalable capacity and resources, up and down
- as the computing requirements change
- Offers different instances' size according to
  - The resources' needs (small, large, and extra large)
  - The high CPU's needs it provides (medium and extra large high CPU instances)
  - High-memory instances (extra large, double extra large, and

quadruple extra large instance)

Amazon EC2 is a widely known example for vendors that provide public cloud services.

Also, Eucalyptus and Open-Nebula are two complementary and enabling technologies for open source cloud tools, which play an invaluable role in infrastructure as a service and in building private, public, and hybrid cloud architecture.

- The Amazon EC2 (Elastic Compute Cloud) is a Web service that allows users to provision new machines into Amazon's virtualized infrastructure in a matter of minutes; using a publicly available API
- EC2 instance is typically a virtual machine with a certain amount of RAM, CPU, and storage capacity.

Amazon EC2 provides its customers with three flexible purchasing models to make it easy for the cost optimization:

- **On-Demand instances**: which allow you to pay a fixed rate by the hourwith no commitment.
- **Reserved instances**: which allow you to pay a low, one-time fee and in turn receive a significant discount on the hourly usage charge for that instance. It ensures that any reserved instance you launch is guaranteed to succeed (provided that you have booked them in advance). This means that users of these instances should not be affected by any transient limitations in EC2 capacity.
- **Spot instances**: which enable you to bid whatever price you want for instance capacity, providing for even greater savings, if your applications have flexible start and end times.

Amazon **Elastic Load Balancer** is another service that helps in building fault-tolerant applications by automatically provisioning incoming application workload across available Amazon EC2 instances and in multiple availability zones.

Private Cloud and Infrastructure Services

A private cloud aims at providing public cloud functionality, but on private resources

- ❖ Maintaining control over an organization's data and resources to meet security and governance's requirements in an organization
- ❖ A highly virtualized cloud data center located inside the organization's firewall
- ❖ Also be a private space dedicated for the company within a cloud vendor's data center designed to handle the organization's workloads

Private clouds exhibit the following characteristics:

- ▫ Allow service provisioning and compute capability for an organization's users in a self-service manner
- ▫ Automate and provide well-managed virtualized environments
- ▫ Optimize computing resources, and servers' utilization
- ▫ Support specific workloads

Examples for vendors and frameworks that provide Iaas in private setups

- ▫ Eucalyptus (elastic utility computing architecture linking your programs to useful systems)

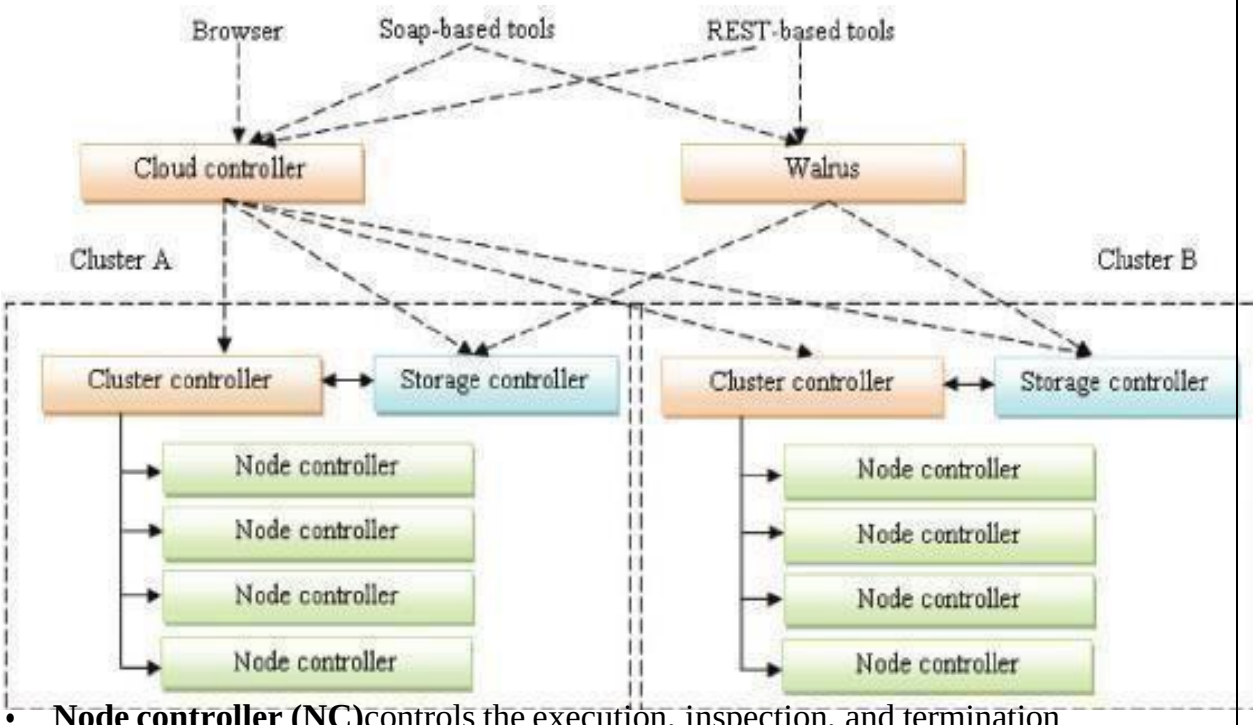
- OpenNebula

Eucalyptus is an open-source infrastructure for the implementation of cloud computing on computer clusters. It is considered one of the earliest tools developed as a surge computing (in which data center's private cloud could augment its ability to handle workload's spikes by a design that allows it to send overflow work to a public cloud) tool. Its name is an acronym for "elastic utility computing architecture for linking your programs to useful systems."

Eucalyptus features :

- Interface compatibility with EC2, and S3 (both Web service and Query/REST [Representational State Transfer] interfaces).
- Simple installation and deployment.
- Support for most Linux distributions (source and binary packages).
- Support for running VMs that run atop the Xen hypervisor or KVM.
- Support for other kinds of VMs, such as VMware, is targeted for future releases.
- Secure internal communication using SOAP (Simple Object Access Protocol) with WS security.
- Cloud administrator's tool for system's management and user's accounting.
- The ability to configure multiple clusters each with private internalnetwork addresses into a single cloud.
- Eucalyptus aims at fostering the research in models for service's provisioning, scheduling, SLA formulation, and hypervisors' portability.

Eucalyptus Architecture:



- **Node controller (NC)**controls the execution, inspection, and termination

of VM instances on the host where it runs.

- **Cluster controller (CC)** gathers information about and schedules VM execution on specific node controllers, as well as manages virtual instance network.
- **Storage controller (SC)** is a put/get storage service that implements Amazon's S3(Simple Storage Service) interface and provides a way for storing and accessing VM images and user data.
- **Cloud controller (CLC)** is the entry point into the cloud for users and administrators. It queries node managers for information about resources, makes high-level scheduling decisions, and implements them by making requests to cluster controllers.
- **Walrus (W)** is the controller component that manages access to the storage services within Eucalyptus. Requests are communicated to Walrus using the SOAP(Simple Object Access Protocol) or REST (Representational State Transfer) based interface

Hybrid Cloud and Infrastructure Services

A third type of cloud setup named Hybrid cloud

- A combination of private/internal and external cloud resources existing together by enabling outsourcing of noncritical services and functions in public cloud and keeping the critical ones internal

Main function of Hybrid cloud is to release resources from a public cloudand handle sudden demand usage called cloud bursting

Distributed Management of Virtualization

- Virtualization needs powerful management capabilities
  Many commercial, open source products and research projects are being developed to dynamically provision virtual machines
  e.g., OpenNebula ,IBM Virtualization Manager, Joyentutilizing the physical infrastructure
- Some commercial and scientific infrastructure cloud computing initiatives provide remote interfaces for controlling and monitoring virtual resources
  e.g., Globus VWS, Eucalyptus and Amazon

The RESERVOIR initiative

- Provides Grid interfaces and protocols enable the required interoperability between the clouds or infrastructure's providers

High Availability

- A system design protocol and an associated implementation ensures a certain absolute degree of operational continuity during a given measurement period
- Availability refers to the ability of a user's community to access the system

- Submitting new work, updating or altering existing work, or collecting the results of the previous work
- Unavailable: A user cannot access the system

Services should be available all the time along with some planned/unplanned downtime according to a certain SLA

- SLA formalizes the service availability objectives and requirements
- The monthly availability or downtime of a service
- To calculate the service's credits to match the billing cycles

- Business critical services are often categorized as high availability services achieving the lowest possible amount of planned and unplanned downtime
- High availability allows virtual machines to automatically be restarted
  - In case of an underlying hardware failure or individual VM failure
  - If one of servers fails, the VMs will be restarted on other virtualized servers in the resource pool restoring the essential services with minimal service interruption

Cloud and Virtualization Standardization Efforts

- Standardization is important to ensure interoperability
- The prevalent standards that make cloud computing and virtualization possible
  - Distributed Management Task Force (DMTF) have produced standards for almost all the aspects of virtualization technology
  - DMTF initiated the VMAN (Virtualization Management) Initiative
    - Delivers broadly supported interoperability and portability standards for managing the virtual computing lifecycle

OVF (Open Virtualization Format)

- VMAN's OVF (Open Virtualization Format) in a collaboration between industry key players: Dell, HP, IBM, Microsoft, XenSource, andVmware.
- OVF specification provides a common format to package and securely distribute virtual appliances across multiple virtualizationplatforms.
- VMAN profiles define a consistent way of managing a heterogeneous virtualized environment
- Standardization effort has been initiated by Open Grid Forum (OGF) through organizing an official new working group to deliver a standard API for cloud IaaS, the Open Cloud Computing Interface Working Group(OCCIWG)

OCCI and OGF

- Another standardization effort has been initiated by Open Grid Forum (OGF) to deliver a standard API for cloud IaaS

- Open Cloud Computing Interface Working Group (OCCI-WG)
  - Dedicated for delivering an API specification for the remote management of cloud computing's infrastructure
  - For allowing the development of interoperable tools for common tasks including deployment, autonomic scaling, and monitoring
  - Covering a high-level functionality required for managing the life-cycle virtual machines/workloads, running on virtualization technologies/containers and supporting service elasticity
- The new API for interfacing IaaS cloud computing facilities will allow
  - Consumers to interact with cloud computing infrastructure on an ad hoc basis
  - Integrators to offer advanced management services
  - Aggregators to offer a single common interface to multiple providers
  - Providers to offer a standard interface that is compatible with the available tools
  - Vendors of grids/clouds to offer standard interfaces for dynamically scalable service's delivery in their products

Virtual Machines Provisioning and Manageability

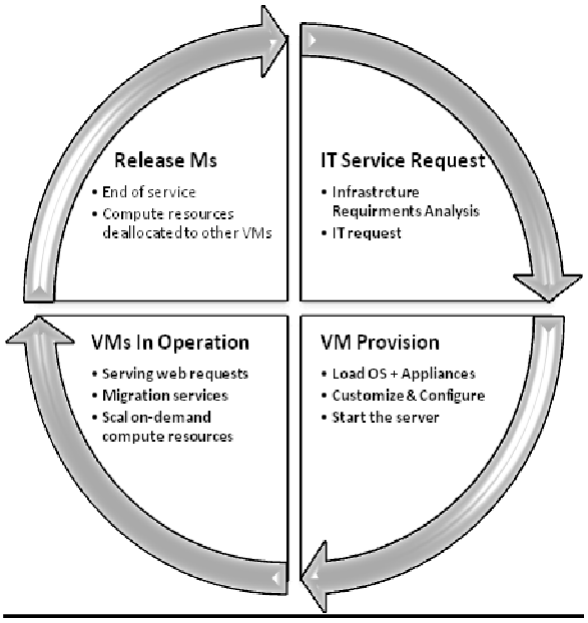
- Typical life cycle of VM and its major possible states of operation, which make the management and automation of VMs in virtual and cloud environmentseasier

Process:

- Steps to Provision VM. Here, we describe the common and normal steps of provisioning a virtualserver:
- Firstly, you need to select a server from a pool of available servers (physical servers with enough capacity) along with the appropriate OS template you need to provision the virtualmachine.
- Secondly, you need to load the appropriate software (operating system you selected in the previous step, device drivers, middleware, and the needed applications for the service required).
- Thirdly, you need to customize and configure the machine (e.g., IP address, Gateway) to configure an associated network and storageresources.
- Finally, the virtual server is ready to start with its newly loadedsoftware

Life cycle of VM and its major possible states of operation

- Starts by a request delivered to the IT department stating the requirement for creating a new server for a particular service
- Processed by the IT administration to start seeing the servers' resource pool matching these resources with the requirements starting the provision of the needed virtual machine
- Once it is provisioned and started it is ready to provide the required service according to an SLA
- A time period after which the VM is being released and resources freed



The common and normal steps of provisioning a virtual server

- Select a server from a pool of available servers (Physical servers with enough capacity) along with the appropriate OS template
- Load the appropriate software
  - Operating system, device drivers, middleware, and the needed applications for the service required
- Customize and configure the machine to configure an associated network and storage resources
  - e.g., IP address, Gateway
  - The virtual server is ready to start with its newly loaded software
- Server provisioning is defining server's configuration

  Based on the organization requirements, a hardware, and software component, processor, RAM, storage, networking, operating system, applications, etc.
- Virtual machines can be provisioned by manually installing an operating system, by using a preconfigured VM template, by cloning an existing VM, or by importing a physical server or a virtual server from another hosting platform

Virtual Machine Migration Services

- **Migration service** is the process of moving a virtual machine from one host server or storage location to another
- Different techniques of VM migration
  - Hot/live migration
  - cold/regular migration
  - Live storage migration of a virtual machine
- In this process, all key machines' components, are completely virtualized
  - e.g., CPU, storage disks, networking, memory
  - Facilitating the entire state of a virtual machine to be captured by a set of easily moved data files

Migration Techniques:

- **Live migration**
  - Also called hot or real-time migration
  - The movement of a virtual machine from one physical host to another while being powered on without any noticeable effect from the end user's point of view (a matter of milliseconds)
  - Facilitates proactive maintenance upon failure
  - The potential problem can be resolved before the disruption of service occurs
  - Used for load balancing
  - Work is shared among computers optimize the utilization of available CPU resources

Live migration's mechanism

- How memory and virtual machine states are being transferred through the network from one host A to another host B
- e.g., the Xen hypervisor
- The process has been viewed as a transactional interaction between the two hosts involved
- **Stage 0**: **Pre-Migration**
  An active virtual machine exists on the physical host A
- **Stage 1**: **Reservation**
  A request is issued to migrate an OS from host A to B
  The necessary resources exist on B and on a VM container of that size
- **Stage 2**: **Iterative Pre-Copy**
  During the first iteration, all pages are transferred from A to B
  - Subsequent iterations copy only those pages dirtied during the previous transfer phase

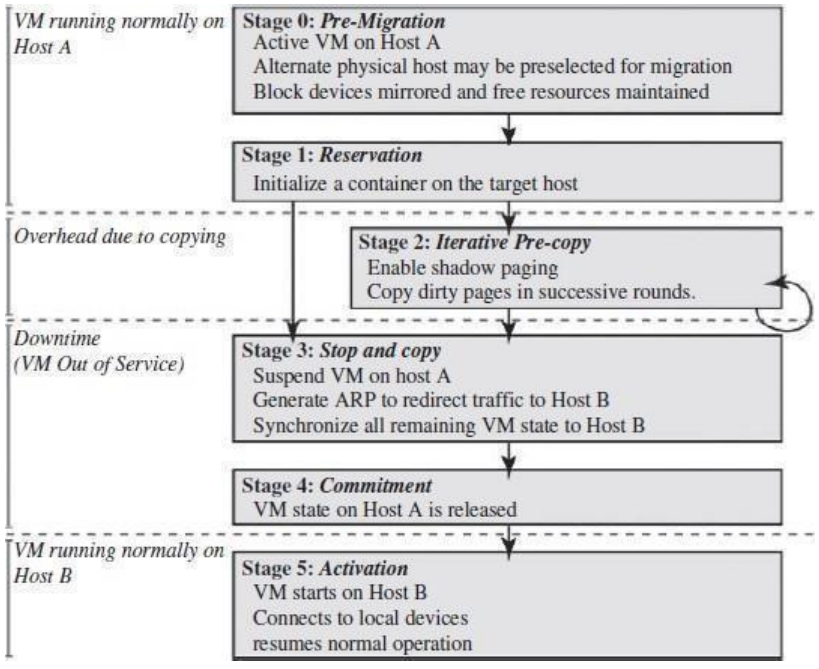
- **Stage 3**: **Stop-and-Copy**

  Running OS instance at A is suspended

  The network traffic is redirected to B

  - CPU state and any remaining inconsistent memory pages are then transferred
  - At the end of this stage, there is a consistent suspended copy of the VM at both A and B.
  - Copy at A is considered primary and is resumed in case of failure

- **Stage 4:Commitment**
  - Host B indicates to A that it has successfully received a consistent OS image
  - Host A acknowledges this message as a commitment of the migration transaction
  - Host A may now discard the original VM
  - Host B becomes the primary host
- **Stage 5: Activation**
  - The migrated VM on B is now activated
  - Post-migration code runs to reattach the device's drivers to the new machine and advertise moved IP addresses



This approach to failure management ensures

- At least one host has a consistent VM image at all times during migration
- The original host remains stable until the migration commits and the VM may be suspended and resumed on that host with no risk of failure

A migration request essentially attempts to move the VM to a new host on any sort of failure, execution is resumed locally aborting the migration

VM Management and Provisioning tools

- Provide the live migration of VM facility
- e.g., VMware VMotion and Citrix XenServerXenMotion

➢ **VMware Vmotion**

- Allows users to automatically optimize and allocate an entire pool of resources for maximum hardware utilization, flexibility, and availability
- To perform hardware's maintenance without scheduled downtime along with migrating virtual machines away from failing or underperforming servers

➢ **Citrix XenServerXenMotion**

- Inherited from the Xen live migrate utility
- Provides the IT administrator with the facility to move a running VM from one Xen Server to another in the same pool without interrupting the service, making it a highly available service
- A good feature to balance the workloads on the virtualized environment

Cold migration

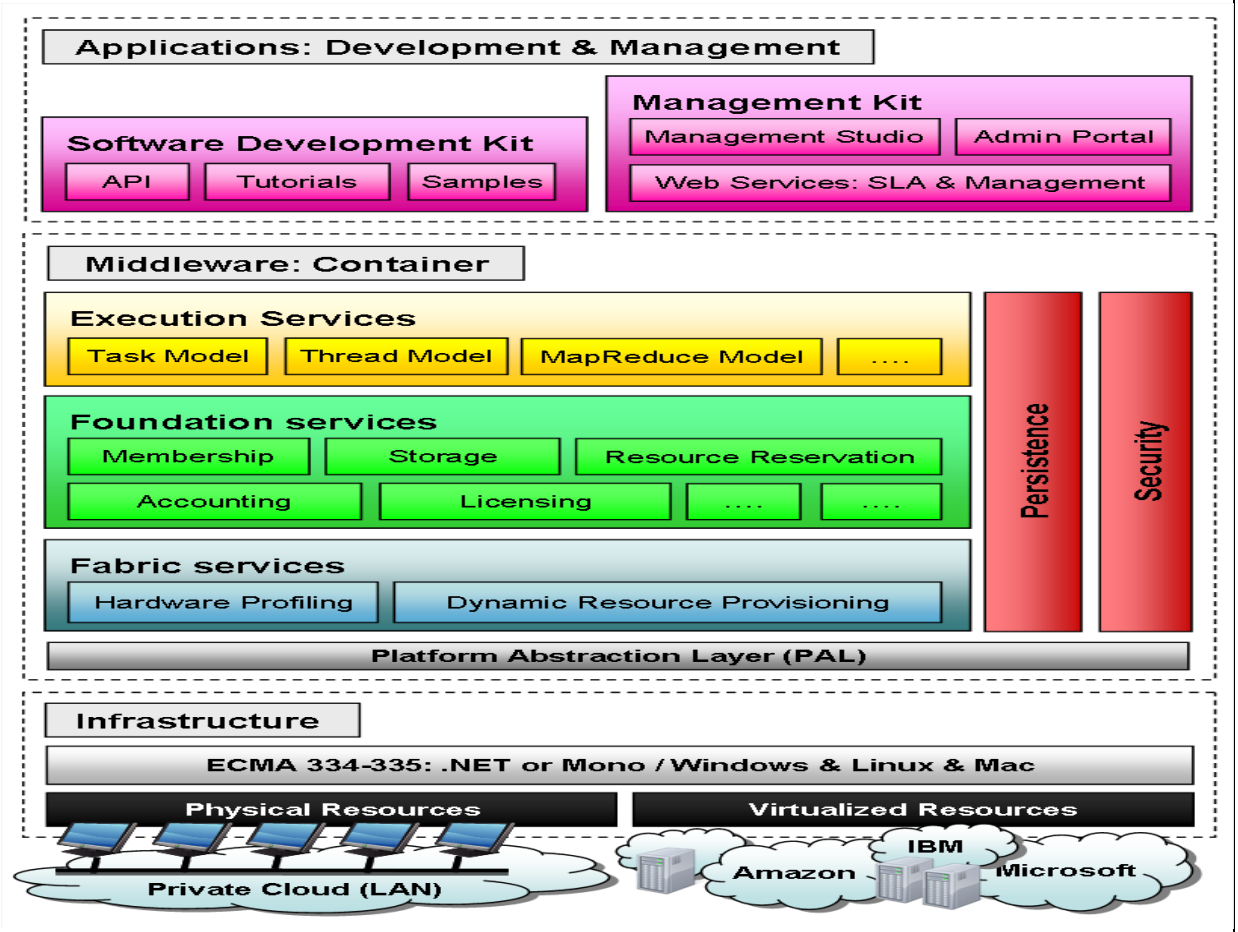
- The migration of a powered-off virtual machine
- Associated disks can be moved from one data store to another
- The virtual machines are not required to be on a shared storage

Differences between **Hot**(Live) migration and **Cold** migration

- Live migration needs a shared storage for virtual machines in the server's pool, but cold migration does not
- In live migration for a virtual machine between two hosts, there would be certain CPU compatibility checks to be applied, in cold migration this checks do not apply

ANEKA CLOUD PLATFORM

Aneka is a software platform and a framework for developing distributed applications on the cloud



- Aneka is essentially an implementation of the PaaS model
  - Provides a runtime environment for executing applications by leveraging the underlying infrastructure of the cloud
  - Developers can express distributed applications
    - By using the API contained in the Software Development Kit (SDK)
    - Or by porting existing legacy applications to the cloud

    Such applications are executed on the Aneka cloud
    - Represented by a collection of nodes connected through the network hosting the Aneka container
  - The container is the building block of the middleware
    - Represents the runtime environment for executing applications
    - Contains the core functionalities of the system

Three classes of services that characterize the container

- **Execution Services**
  - Responsible for scheduling and executing applications
- **Foundation Services**
  - The core management services of the Aneka container
  - In charge of metering applications, allocating resources for execution, managing the collection of available nodes, and keeping the services registry updated
- **Fabric Services**
  - Constitute the lowest level of the services stack of Aneka
  - Provide access to the resources managed by the cloud
  - The Resource Provisioning Service enables horizontal scaling in the cloud
  - Horizontal scaling is the process of adding more computing nodes to a system
  - Vertical scaling is the process of increasing the computing capability of a single computer resource
  - Resource provisioning makes Aneka elastic
  - Allows to grow or to shrink dynamically to meet the QoS requirements of applications

The container relies on a platform abstraction layer

- Interfaces it with the underlying host whether this is a physical or a virtualized resource
- Makes the container portable over different runtime environments
  - Feature an implementation of the ECMA 334 and ECMA 335 specifications
  - e.g., the .NET framework or Mono

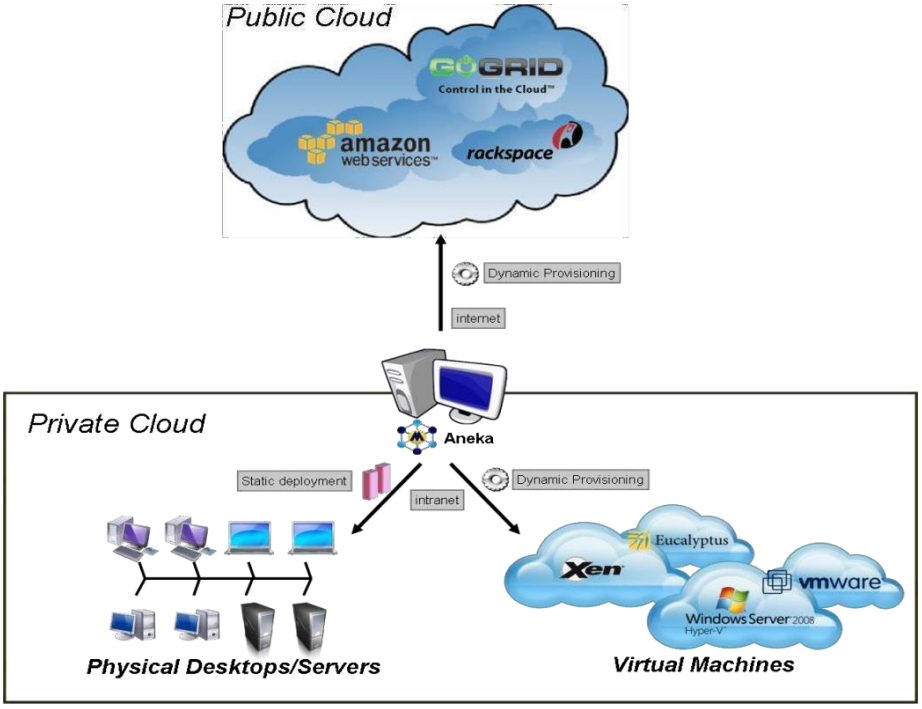
Aneka also provides a tool for managing the cloud

- Allowing administrators to start, stop, and deploy instances of the container on new resources and then reconfigure them dynamically to alter the behavior of the cloud

Aneka Resource Provisioning Service

- Cloud computing has the ability to automatically scale out
  - Based on demand and users' quality of service requests
- Aneka is a PaaS
  - Features multiple programming models allowing developers to easily build their distributed applications
  - Provides resource provisioning facilities in a seamless and dynamic fashion
  - Achieved by means of the resource provisioning framework
- A typical scenario that a medium or large enterprise may encounter
  - Combines privately owned resources with public rented resources to dynamically increase the resource capacity to a larger scale

Private resources identify computing and storage elements kept in the premises. They share similar internal security and administrative policies



- Aneka identifies two types of private resources
  - Static and dynamic resources
  - **Static resources** are constituted by existing physical workstations and servers
    - May be idle for a certain period of time
    - Their membership to the Aneka cloud is manually configured by administrators
    - Does not change over time
  - **Dynamic resources** are mostly represented by virtual instances that join and leave the cloud
    - Controlled by resource pool managers that provision and release them when needed
- Public resources reside outside the boundaries of the enterprise
  - Provisioned by establishing a service-level agreement with the external provider

  Two classes: on-demand and reserved resources
  - **On-demand** resources are dynamically provisioned by resource pools for a fixed amount of time, e.g., an hour
    - With no long-term commitments
    - On a pay-as-you-go basis
  - **Reserved resources** are provisioned in advance by paying a low, one-time fee
    - Mostly suited for long-term usage
    - Actually the same as static resources
    - No automation is needed in the resource provisioning service to manage them

CHALLENGES AND RISKS

Despite the initial success and popularity of the cloud computing paradigm and the extensive availability of providers and tools, a significant number of challenges and risks are inherent to this new model of computing. Providers, developers, and end users must consider these challenges and risks to take good advantage of cloud computing. Issues to be faced include user privacy, data security, data lock-in, availability of service, disaster recovery, performance, scalability, energy-efficiency, andprogrammability.

Security, Privacy, and Trust: Security and privacy affect the entire cloud computing stack, since there is a massive use of third-party services and infrastructures that are used to host important data or to perform critical operations. In this scenario, the trust toward providers is fundamental to ensure the desired level of privacy for applications hosted in the cloud. Legal and regulatory issues also need attention. When data are moved into the Cloud, providers may choose to locate them anywhere on the planet. The physical location of data centers determines the set of laws that can be applied to the management of data. For example, specific cryptography techniques could not be used because they are not allowed in some countries. Similarly, country laws can impose that sensitive data, such as patient health records, are to be stored within nationalborders.

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

The answer to this concern is standardization. In this direction, there are efforts to create open standards for cloud computing. The Cloud Computing Interoperability Forum (CCIF) was formed by organizations such as Intel, Sun, and Cisco in order to "enable a global cloud computing ecosystem whereby organizations are able to seamlessly work together for the purposes for wider industry adoption of cloud computing technology." The development of the Unified Cloud Interface (UCI) by CCIF aims at creating a standard programmatic point of access to an entire cloud infrastructure. In the hardware virtualization sphere, the Open Virtual Format (OVF) aims at facilitating packing and distribution of software to be run on VMs so that virtual appliances can be made portable—that is, seamlessly run on hypervisor of differentvendors.

Availability, Fault-Tolerance, and Disaster Recovery: It is expected that users willhave certain expectations about the service level to be provided once their applications are moved to the cloud. These expectations include availability of the service, its overall performance, and what measures are to be taken when something goes wrong in the system or its components. In summary, users seek for a warranty before they can comfortably move their business to the cloud. SLAs, which include QoS requirements, must be ideally set up between customers and cloud computing providers to act as warranty. An SLA specifies the details of the service to be provided, including availability and performance guarantees. Additionally, metrics must be agreed upon by all parties, and penalties for violating the expectations must also beapproved.

Resource Management and Energy-Efficiency: One important challenge faced by providers of cloud computing services is the efficient management of virtualized resource pools. Physical resources such as CPU cores, disk space, and network bandwidth must be sliced and shared among virtual machines running potentially heterogeneous workloads. The multi-dimensional nature of virtual machines complicates the activity of finding a good mapping of VMs onto available physical hosts while maximizing user utility. Dimensions to be considered include: number of CPUs, amount of memory, size of virtual disks, and network bandwidth. Dynamic VM mapping policies may leverage the ability to suspend, migrate, and resume VMs as an easy way of preempting low-priority allocations in favor of higher-priority ones. Migration of VMs also brings additional challenges such as detecting when to initiate a migration, which VM to migrate, and where to migrate. In addition, policies may take advantage of live migration of virtual machines to relocate data center load without significantly disrupting running services. In this case, an additional concern is the trade-off between the negative impact of a live migration on the performance and stability of a service and the benefits to be achieved with that migration. Another challenge concerns the outstanding amount of data to be managed in variousVM management activities. Such data amount is a result of particular abilities of virtual machines, including the ability of traveling through space (i.e., migration) and time (i.e., checkpointing and rewinding), operations that may be required in load balancing, backup, and recovery scenarios. In addition, dynamic provisioning of new VMs and replicating existing VMs require efficient mechanisms to make VM block storage devices (e.g., image files) quickly available at selected hosts. Data centers consumer large amounts of electricity. According to a data published byHP[4], 100 server racks can consume 1.3MWof power and another 1.3 MW are required by the cooling system, thus costing USD 2.6 million per year. Besides the monetary cost, data centers s ignificantly impact the environment in terms of CO2 emissions from the coolingsystems

UNIT- IV

Software as a Service (SAAS) &Data Security in the Cloud: Software as a Service SAAS), Google App Engine – Centralizing Email Communications- Collaborating via Web-Based Communication Tools-An Introduction to the idea of Data Security.

SAAS

- SaaS is a model of software deployment where an application is hosted as a service provided to customers across the Internet.
- Saas alleviates the burden of software maintenance/support but users relinquish control over software versions and requirements
- SaaS is at the highest layer and features a complete application offered as a service, on-demand, via multitenancy — meaning a single instance of the software runs on the provider's infrastructure and serves multiple client organizations.

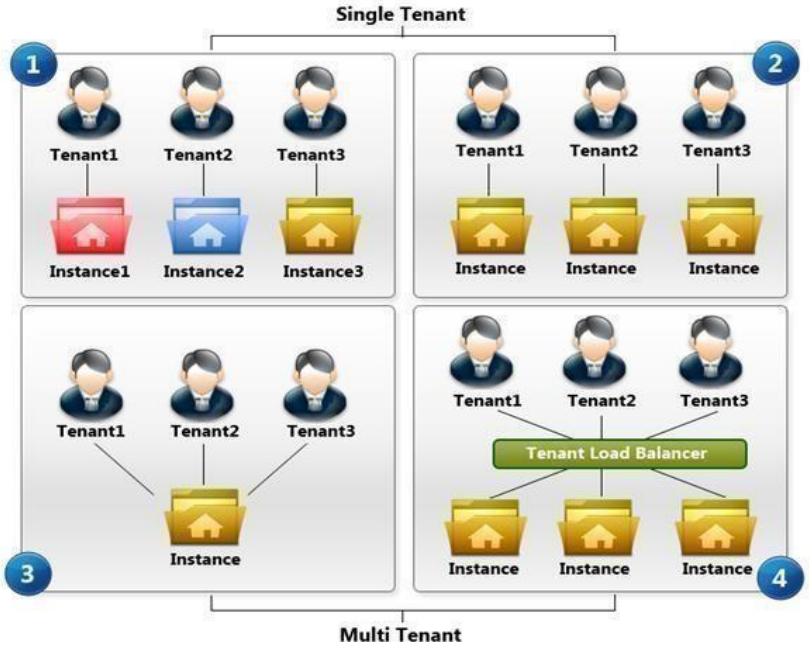
SaaS Maturity Model

Level 1: Ad-Hoc/Custom – One Instance per customer

Level 2: Configurable per customer

Level 3: configurable & Multi-Tenant-Efficient

Level 4: Scalable, Configurable & Multi-Tenant-Efficient



Single Tenant

In a single tenant model each of the tenants will get their own respective instances. There is absolutely no sharing of anything (code, DB, etc.). Each time a new customer (tenant) is added a new (logical) hardware is provisioned and new instance of the product is setup in the allocated environment. This could also be a logical separation of the hardware in the form of

separate web sites within the same IIS server. Single tenant model is also commonly referred as Hosted model.

Maturity Level 1 and 2 falls under single tenant model. In both the levels the tenant gets their own instance of the software. However, the primary difference between these levels is that in Level 1 you may even have a customized code base for that particular tenant. Therefore, you may even end up having different code bases for each tenant (as depicted with multiple colors in the above picture). However, from a Tenant's perspective it's immaterial whether it's Level 1 or 2 as long as the software performs the expected functions.

Advantages

- Takes less time to roll-out in to SaaS model, since the product does not require going through any changes to support SaaS model.
- Overall SaaS transition complexity and cost is going to be less.
- Does not require any SaaS Architecture/Engineering expertise. All that is required would be expertise on the hardware front, which is any addressed by IaaS vendors.
- Level 1 offers a unique advantage of the ability to provide customized versions to your client. However, this could also be viewed as a disadvantage as you will have to keep maintaining several versions of your product.
- Supports non-web native applications. For example, you can use windows citrix to deliver a desktop or client/server application (ex: applications developed in VB) in a SaaS model.
- Certain market/customer base does not like the idea of multi-tenancy. For example: a banking customer will not like the idea of sharing their data/environment with other banking organizations. In some cases the security compliance also does not allow sharing. In these instances Single Tenant wins over multi-tenant.

Disadvantages

- Maintenance efforts are going to be huge as you will have to maintain multiple code base/environments. For example, if you are going to make a fix then you will have to roll-it out 'n' number of times, where n is the number of tenants supported.
- Operational costs are going to be extremely high, particularly over the long run.

Multi-Tenant

Multi-Tenancy is an architecture capability that allows an application/product to recognize tenants (tenants could be users, group of users or organizations) and exhibit functionalities as per the configuration set for the tenants. An on-premise application is typically designed to work for a single organization. Therefore, the very concept of making an on-premise application realize/operate in the context of a particular tenant is a big change. Every single functionality in the product has to be tweaked in order to achieve multi-tenancy. Segregation at the data layer is another big challenge. There are various degrees of multi-tenancy that can be supported. The complexity/effort also depends on the multi-tenancy degree that you are choosing to go with.

Maturity Level 3 and 4 fall under the multi-tenant model. Level 3 and 4 differ only in the level of scalability they can offer. Level 3 is more of a scale-up solution where you can upgrade the hardware to increase the number of tenants that can be supported. Level 4

supports scale-out solution where multiple hardwares can be added so that a load balanced environment can be created. As you can imagine, the decision between Level 3 and Level 4 lies with the amount of tenants that you are expecting to have on the system.

Advantages

- Facilitates a cost effective way of delivering SaaS solution.
- Huge savings in operational cost, particularly over the long run. This in turn will allow the ISVs to offer an attractive pricing to their customers, therefore creating a snow ball effect.
- Enables to achieve higher levels of customer satisfaction.
- Roll-out of upgrades/fixes is much easier.
- Easier to adopt/implement best practices in the product features as you will be concentrating on only one version of the product.
- The design principles of a multi-tenant system offer a high level of maintainability. For example, if a customer requests for few additional fields in one of the pages you can easily add them through custom field's module.
- Scalability to expand the number of tenants. For example: Salesforce.com has more than 100,000 tenants running on their multi-tenant system. Can you even imagine how this will work in a single tenant model?

Disadvantages

- Due to the amount of changes that has to happen in the product it takes a while to roll-out the solution.
- Initial investment to enable multi-tenancy is high. While you will recover this cost in the form of savings in operational expenses it still requires that initial investment and as well the risk involved in achieving the expected business growth.
- Requires SaaS architecture expertise, which may not be within the company. Will have to find a strong SaaS partner to guide you in the transition process.

As you can see each model has its own advantage and disadvantages. Hence, it's extremely important to think through for a given business/domain/product to understand if multi-tenancy is really required or not. In my next blog I will talk about how to objectively make this decision.

SaaS INTEGRATION PRODUCTS AND PLATFORMS

- Cloud-centric integration solutions are being developed and demonstrated for showcasing their capabilities for integrating enterprise and cloudapplications.
- Composition and collaboration will become critical and crucial for the mass adoption of clouds

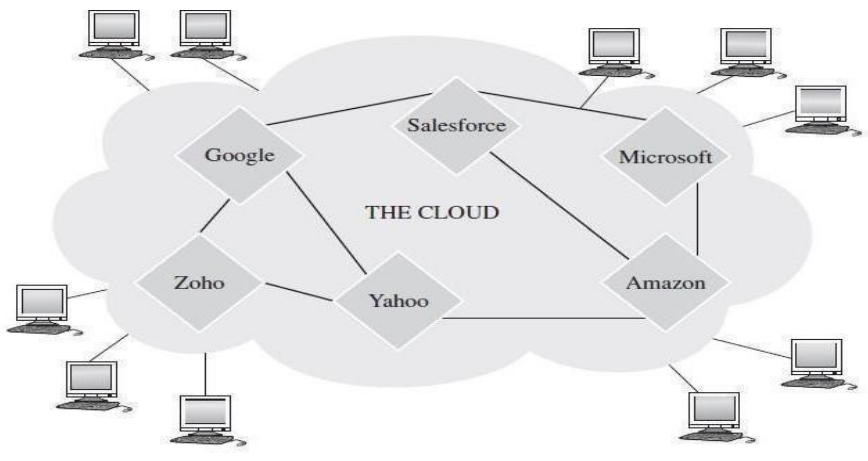


FIGURE 3.4. The Smooth and Spontaneous Cloud Interaction via Open Clouds.

Jitterbit:

- Jitterbitis a fully graphical integration solution that provides users a versatileplatform
- suite of productivity tools to reduce the integration effortssharply.
- Jitterbit can be used standalone or with existing EAIinfrastructures
- Help us quickly design, implement, test, deploy, and manage the integrationprojects

Two major components:

❖ Jitterbit IntegrationEnvironment

- An intuitive point-and-click graphical UI that enables to quickly configure, test, deploy and manage integration projects on the Jitterbitserver.

❖ Jitterbit IntegrationServer

- A powerful and scalable run-time engine that processes all the integration operations, fully configurable and manageable from the Jitterbitapplication.

Linkage with On premise and on demand Applications

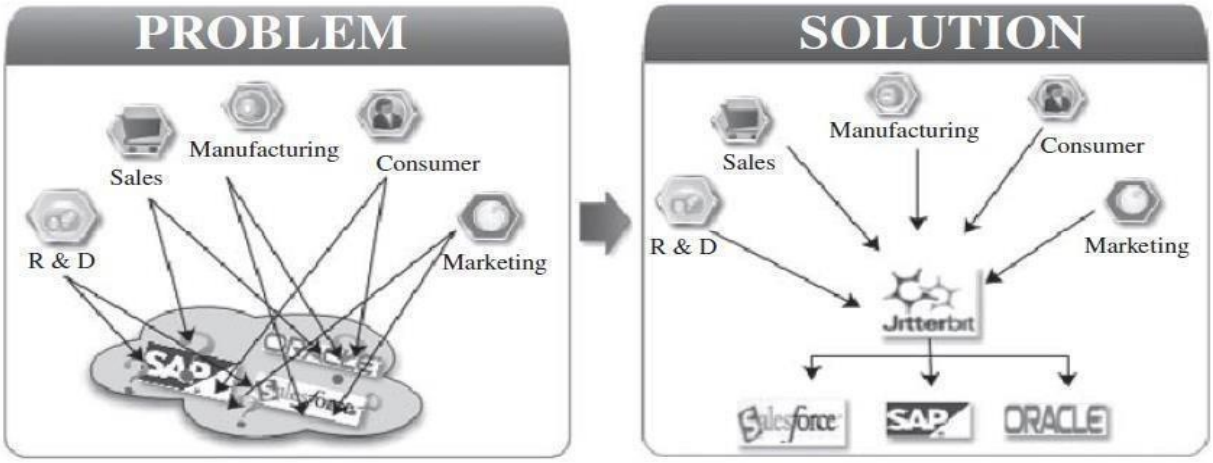


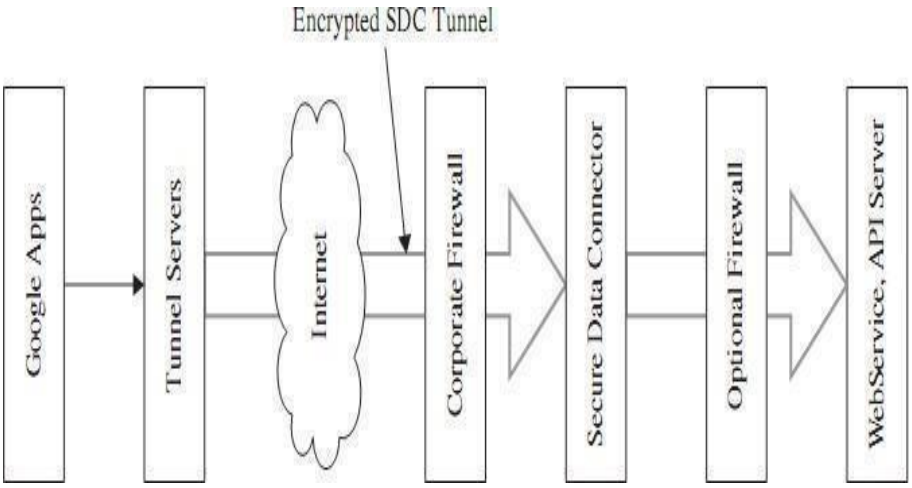
FIGURE 3.5. Linkage of On-Premise with Online and On-Demand Applications.

Google APP Engine

- The app engine is a Cloud-based platform, is quite comprehensive and combines infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS).
- The app engine supports the delivery, testing and development of software on demand in a Cloud computing environment that supports millions of users and is highlyscalable.
- The company extends its platform and infrastructure to the Cloud through its app engine. It presents the platform to those who want to develop SaaS solutions at competitivecosts
- It is a platform for hosting web applications in Google managed data centers. It is cloud computing technology which virtualizes applications across multiple servers and data centers.
- Running your web application in Google infrastructure
- Support different runtime environments

Java (JRE 6 with limitation, Servlet 2.5, JDO, JPA)
Python (2.5.2)

- Apps run in sandbox.
- Automatic scaling and load balancing
- No server restart, no network issues



- The SDC constructs an encrypted connection between the data source and Google Apps. As long as the data source is in the Google Apps domain to the Google tunnel protocol servers, when the user wants to get the data, he/she will first send an authorized data requests to Google Apps, which forwards the request to the tunnel server.
- The tunnel servers validate the request identity. If the identity is valid, the tunnel protocol allows the SDC to set up a connection, authenticate, and encrypt the data that flows across the Internet. At the same time, the SDC uses resource rules to validate whether a user is authorized to access a specified resource.
- When the request is valid, the SDC performs a network request. The server validates the signed request, checks the credentials, and returns the data if the user is authorized From the perspective of cloud storage services, data integrity depends on the security of operations while in storage in addition to the security of the uploading and downloading sessions. The uploading session can only ensure that the data received by the cloud storage is the data that the user uploaded; The downloading session can guarantee the data that the user retrieved is the data cloud storage recorded. Unfortunately, this procedure applied on cloud storage services cannot guarantee data integrity. First, assume that Alice, a company CFO, stores the company financial data at a cloud storage service provided by Eve. And then Bob, the company administration chairman, downloads the data from the cloud.

There are three important concerns in this simple procedure:

1. Confidentiality. Eve is considered as an untrustworthy third party, Alice and Bob do not want reveal the data to Eve.

2. Integrity. As the administrator of the storage service, Eve has the capability to play with the data in hand. How can Bob be confident that the data he fetched from Eve are the same as what was sent by Alice? Are there any measures to guarantee that the data have not been tampered by Eve?

3. Repudiation. If Bob finds that the data have been tampered with, is there any evidence for him to demonstrate that it is Eve who should be responsible for the fault? Similarly, Eve also needs certain evidence to prove her innocence.

GoogleAPPEngine

Solutions for Missing Link

- Third Authority Certified(TAC)
- Secret Key Sharing(SKS)

Four Solutions

- Neither TAC nor SKS
- With SKS but without TAC
- With TAC but without SKS
- With Both TAC and SKS

Google is a leader in web-based applications,

so it's not surprising that the company also offers cloud development services.

- These services come in the form of the Google App Engine, which enables developers to build their own web applications utilizing the same infrastructure that powers Google's powerfulapplications.
- The Google App Engine provides a fully integrated application environment. Using Google's development tools and computing cloud, App Engine applications are easy to build, easy to maintain, and easy to scale. All you have todo

Features of App Engine

- These are covered by the depreciation policy and the service-level agreement of the app engine. Any changes made to such a feature are backward-compatible and implementation of such a feature is usually stable. These include data storage, retrieval, and search; communications; process management; computation; app configuration and management.

- Data storage, retrieval, and search include features such as HRD migration tool, Google Cloud SQL, logs, datastore, dedicated Memcache, blobstore, Memcache andsearch.

- Communications include features such as XMPP. channel, URL fetch, mail, and Google Cloud Endpoints.

- Process management includes features like scheduled tasks and taskqueue
- Computation includesimages.
- App management and configuration cover app identity, users, capabilities, traffic splitting, modules, SSL for custom domains, modules, remote access, andmultitenancy

Centralizing email Communications

- The key here is to enable anywhere/anytime access toemail.
- Precloud computing, your email access was via a single computer, which also stored all your email messages. For this purpose, you probably used a program like Microsoft Outlook or Outlook Express, installed on your homecomputer.
- To check your home email from work, it took a bit of juggling and perhaps the use of yourISP'semailaccesswebpage.Thatwebpagewasneverinsyncwiththemessages

  on your home PC, of course, which is just the start of the problems with trying to communicate in this fashion.
- A better approach is to use a web-based email service, such as Google's Gmail (mail.google.com), Microsoft's Windows Live Hotmail (mail.live.com), or Yahoo! Mail (mail.yahoo.com). These services place your email inbox in the cloud; you can access it from any computer connected to theInternet.

1. Cloud computing for community: It has tremendous benefits for the entire community, from neighborhood groups to sports teams to school organizations. Any time any groups of people in the community need to communicate and collaborate;web-based applications are the way to go.

2. Communicating Across the Community: One of the key components of any community collaboration is communication. Many community activities are undertaken by people in their spare timeoutside of normal work and home activities. Therefore, they might be communicating during office hours on their work computer, after hours on their home computer, or during any spare moment. Programs can be accessed from any computer connected to the Internet.

3. Collaborating on Schedules :It comes to coordinating multiple individuals or families in a community activity; you have your work cut out for you. Whether it's a youth sports team, community organization, school event, or some community event, trying to line up who's free and who's not on a given evening takes a lot of effortunless, that is, you're using webbased scheduling tools.

Collaborating via Web-Based Communication Tools GMAIL

- Gmail offers a few unique features that set it apart from the web-based emailcrowd.
- First, Gmail doesn't use folders. With Gmail you can't organize your mail into folders, as you can with the otherservices.
- Instead, Gmail pushes the search paradigm as the way to find the messages you want— not a surprise, given Google's search-centric businessmodel.
- Gmail does, however, let you "tag" each message with one or more labels. This has the effect of creating virtual folders, as you can search and sort your messages by any oftheir labels.
- In addition, Gmail groups together related email messages in what Google calls conversations

Yahoo! Mail Yahoo! Mail (mail.yahoo.com)

- isanother web mail service, provided by the popular Yahoo! searchsite.
- The basic Yahoo! Mail is free and can be accessed from any PC, using any web browser.
- Yahoo! also offers a paid service called Yahoo! Mail Plus that lets you send larger messages and offers offline access to your messages via POP emailclients

Web Mail Services

- AOL Mail(mail.aol.com)
- BigString (www.bigstring.com)E
- xcite Mail(mail.excite.com)
- FlashMail(www.flashmail.com)
- GMX Mail(www.gmx.com)
- Inbox.com(www.inbox.com)
- Lycos Mail(mail.lycos.com)
- Mail.com(www.mail.com)
- Zoho Mail(zoho.mail.com)

An Introduction to Data Security

- Information in a cloud environment has much more dynamism and fluidity than information that is static on a desktop or in a networkfolder
- Nature of cloud computing dictates that data are fluid objects, accessible from a multitude of nodes and geographic locations and, as such, must have a data security methodology that takes this into account while ensuring that this fluidity is notcompromised

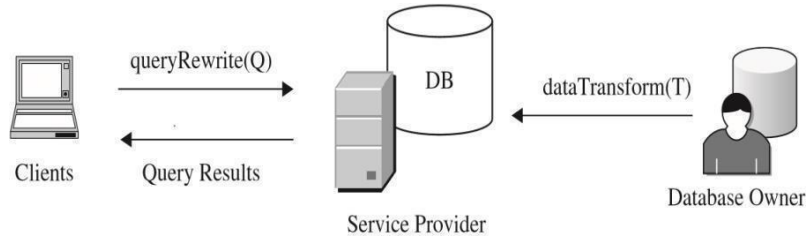
- The idea of content-centric or information-centric protection, being an inherent part of a data object is a development out of the idea of the "de-perimerization" of theenterprise.
- This idea was put forward by a group of Chief Information Officers (CIOs) who formed an organization called the JerichoForum

TECHNOLOGIES FOR DATA SECURITY IN CLOUD COMPUTING

Unique issues of the cloud data storage platform from a few different perspectives

- Database Outsourcing and Query Integrity Assurance

- Storing data into and fetching data from devices and machines behind a cloud are essentially a novel form of database outsourcing



- Data Integrity in Untrustworthy Storage

- The fear of losing data or data corruption
- Relieve the users' fear by providing technologies that enable users to check the integrity of their data

Web-Application-Based Security

- Once the dataset is stored remotely, a Web browser is one of the most convenient approaches that end users can use to access their data on remote services
- Web security plays a more important role for cloud computing

- Multimedia Data Security

- With the development of high-speed network technologies and large bandwidth connections, more and more multimedia data are being stored and shared in cyber space
- The security requirements for video, audio, pictures, or images are different from other applications

CLOUD COMPUTING AND IDENTITY

Digital identity

- holds the key to flexible data security within a cloudEnvironment
- A digital identity represents who we are and how we interact with otherson-line.
- **Access, identity, and risk** are three variables that can become inherently connected when applied to the security of data, because access and risk are directly proportional: As access increases, so then risk to the security of the dataincreases.
- Access controlled by identifying the actor attempting the access is the most logical manner of performing thisoperation.
- Ultimately, digital identity holds the key to securing data, if that digital identity can be programmatically linked to security policies controlling the post-access usage ofdata.

Identity, Reputation, and Trust

- Reputation is a real-world commodity; that is a basic requirement of human-to-human relationships
- Our basic societal communication structure is built upon the idea of reputation andtrust.
- Reputation and its counter value, trust, is easily transferable to a digitalrealm:
  - oeBay, for example, having partly built a successful business model on the strength of a ratings system, builds up the reputation of its buyers and sellers through successful (or unsuccessful) transactions.
- These types of reputation systems can be extremely useful when used with a digital identity.
- They can be used to associate varying levels of trust with that identity, which in turn can be used to define the level (granular variations) of security policy applied to data resources that the individual wishes toaccess

User-Centric Identity:

- Digitalidentitiesareamechanismforidentifyinganindividual,particularlywithinacloud environment ; identity ownership being placed upon the individual is known as user- centricidentity
- It allows users to consent and control how their identity (and the individualidentifiers making up the identity, the claims) isused.
  - This reversal of ownership away from centrally managed identity platforms(enterprise- centric) has manyadvantages.
- This includes the potential to improve the privacy aspects of a digital identity, by givingan individual the ability to apply permission policies based on their identity and to control which aspects of that identity aredivulged
  - An identity may be controllable by the end user, to the extent that the user can then decide what information is given to the party relying on theidentity

Information Card:

- Information cards permit a user to present to a Web site or other service (relying party) one or more claims, in the form of a software token, which may be used to uniquely identify thatuser.

- They can be used in place of user name/ passwords, digital certificates, and other identification systems, when user identity needs to be established to control access to a Web site or other resource, or to permit digitalsigning

Information cards are part of an identity meta-system consisting of:

- 1. **Identity providers (IdP**), who provision and manage information cards,with specific claims, tousers.
- 2. **Users** who own and utilize the cards to gain access to Web sites and other resources that support information cards.
- **An identity selector/service**, which is a piece of software on the user's desktop or in the cloud that allows a user to select and manage theircards.
- 4. **Relying parties.** These are the applications, services, and so on, that can use an information card to authenticate a person and to then authorize an action such as logging onto a Web site, accessing a document, signing content, and soon

Each information card is associated with a set of claims which can be used toidentify the user. These claims include identifiers such as name, email address,post code

Using Information Cards to Protect Data

- Information cards are built around a set of open standards devised by a consortium that includes Microsoft, IBM, Novell, and soon.
- The original remit of the cards was to create a type of single sign on system for the Internet, to help users to move away from the need to remember multiplepasswords.
- However, the information card system can be used in many moreways.
- Because an information card is a type of digital identity, it can be used in the same way that other digital identities can be used.

For example, an information card can be used to digitally sign data and content and to control access to data and content. One of the more sophisticated uses of an information card is the advantage given to the cards by way of the claims system.

Cloud Computing and Data Security Risk

- Cloud computing is a development that is meant to allow more open accessibility and easier and improved datasharing.
- Data are uploaded into a cloud and stored in a data center, for access by users from that data center; or in a more fully cloud-based model, the

data themselves are created in the cloud and stored and accessed from the cloud (again via a datacenter).

- The most obvious risk in this scenario is that associated with the storage of that data. A user uploading or creating cloud-based data include those data that are stored and maintained by a third-party cloud provider such as Google, Amazon, Microsoft, and so on.

This action has several risks associated with it:

- Firstly, it is necessary to protect the data during upload into the data center to ensure that the data do not get hijacked on the way into thedatabase.
- Secondly, it is necessary to the stores the data in the data center to ensure that they are encrypted at all times.
- Thirdly, and perhaps less obvious, the access to those data need to be controlled; this control should also be applied to the hosting company, including the administrators of the data center.
- In addition, an area often forgotten in the application of security to a data resource is the protection of that resource during itsuse

Data security risks are compounded by the open nature of cloud computing.

- Access control becomes a much more fundamental issue in cloud-based systems because of the accessibility of thedata
- Information-centric access control (as opposed to access control lists) can help to balance improved accessibility with risk, by associating access rules with different data objects within an open and accessible platform, without losing the Inherent usability of that platform
- A further area of risk associated not only with cloud computing, but also with traditional network computing, is the use of content afteraccess.
- The risk is potentially higher in a cloud network, for the simple reason that the information is outside of your corporatewalls

Data-centric mashups are those

- that are used to perform business processes around data creation and dissemination—by their very nature, can be used to hijack data, leaking sensitive information and/or affecting integrity of thatdata
- Cloud computing, more than any other form of digital communication technology, has created a need to ensure that protection is applied at the inception of the information, in a content centric manner, ensuring that a security policy becomes an integral part of that data throughout its lifecycle.

Encryption

- isa vital component of the protection policy, but further controls over the access of that data and on the use of the data must bemet.
- In the case of mashups, the controlling of access to data resources, can

help toalleviate the security concerns by ensuring that mashup access is authenticated.

- Linking security policies, as applied to the use of content, to the access control method offer a way of continuing protection of data, post access and throughout the life cycle; this type of data security philosophy must be incorporated into the use of cloud computing to alleviate security risks.

UNIT- V

SLA Management in cloud computing: Traditional Approaches to SLO Management, Types of SLA, Life Cycle of SLA, SLA Management in Cloud.

SLA MANAGEMENT IN CLOUD COMPUTING

In the early days of web-application deployment, performance of the application at peak load was a single important criterion for provisioning server resources [1]. Provisioning in those days involved deciding hardware configuration, determining the number of physical machines, and acquiring them upfront so that the overall business objectives could be achieved. The web applications were hosted on these dedicated individual servers within enterprises' own server rooms. These web applications were used to provide different kinds of e-services to various clients. Typically, the service-level objectives (SLOs) for these applications were response time and throughput of the application end-user requests. The capacity buildup was to cater to the estimated peak load experienced by the application. The activity of determining the number of servers and their capacity that could satisfactorily serve the application end-user requests at peak loads is called capacity planning [1]. An example scenario where two web applications, application A and application B, are hosted on a separate set of dedicated servers within the enterprise-owned server rooms is shown in Figure 16.1. The planned capacity for each of the applications to run successfully is three servers. As the number of web applications grew, the server rooms in the organization became large and such server rooms were known as data centers. These data centers were owned and managed by the enterprises themselves.

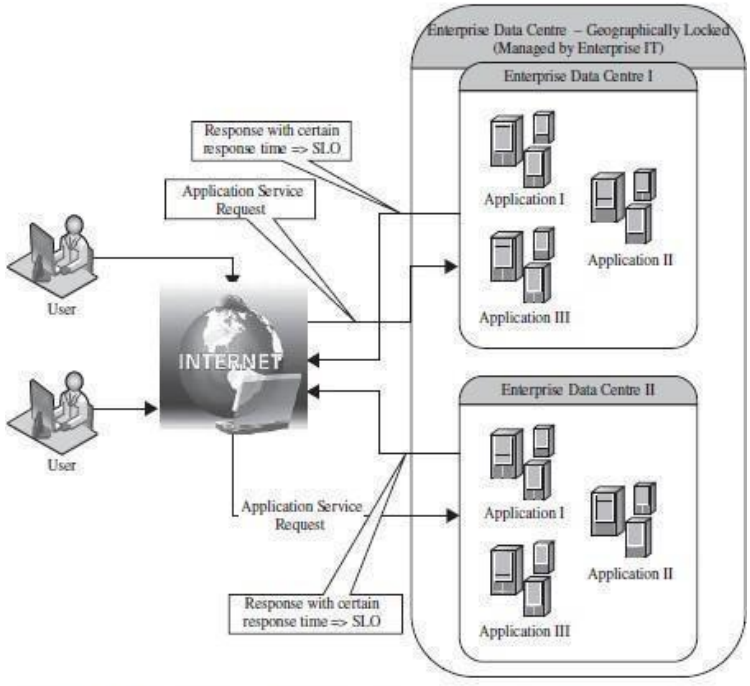


FIGURE 16.1. Hosting of applications on servers within enterprise's data centers.

TRADITIONAL APPROACHES TO SLO MANAGEMENT

Traditionally, load balancing techniques and admission control mechanisms have been used to provide guaranteed quality of service (QoS) for hosted web applications. These mechanisms can be viewed as the first attempt towards managing the SLOs. In the following subsections we discuss the existing approaches for load balancing and admission control for ensuring QoS.

16.2.1 Load Balancing The objective of a load balancing is to distribute the incoming requests onto a set of physical machines, each hosting a replica of an application, so that the load on the machines is equally distributed [4]. The load balancing algorithm executes on a physical machine that interfaces with the clients. This physical machine, also called the front-end node, receives the incoming requests and distributes these requests to different physical machines for further execution.

This set of physical machines is responsible for serving the incoming requests and are known as the back-end nodes.

TYPES OF SLA:

Service-level agreement provides a framework within which both seller and buyer of a service can pursue a profitable service business relationship. It outlines the broad understanding between the service provider and the service consumer for conducting business and forms the basis for maintaining a mutually beneficial relationship. From a legal perspective, the necessary terms and conditions that bind the service provider to provide services continually to the service consumer are formally defined in SLA.

SLA can be modeled using web service-level agreement (WSLA) language specification [7]. Although WSLA is intended for web-service-based applications, it is equally applicable for hosting of applications. Service-level parameter, metric, function, measurement directive, service-level objective, and penalty are some of the important components of WSLA and are described in Table 16.1.

TABLE 16.1. Key Components of a Service-Level Agreement

Service-Level Parameter	Describes an observable property of a service whose value is measurable.
Metrics	These are definitions of values of service properties that are measured from a service-providing system or computed from other metrics and constants. Metrics are the key instrument to describe exactly what SLA parameters mean by specifying how to measure or compute the parameter values.
Function	A function specifies how to compute a metric's value from the values of other metrics and constants. Functions are central to describing exactly how SLA parameters are computed from resource metrics.
Measurement directives	These specify how to measure a metric.

There are two types of SLAs from the perspective of application hosting. These are described in detail here.

Infrastructure SLA. The infrastructure provider manages and offers guarantees on availability of the infrastructure, namely, server machine, power, network connectivity, and so on. Enterprises manage themselves, their applications that are deployed on these server machines. The machines are leased to the customers and are isolated from machines of other customers. In such dedicated hosting environments, a practical example of service-level guarantees offered by infrastructure providers is shown in Table 16.2.

Application SLA. In the application co-location hosting model, the server capacity is available to the applications based solely on their resource demands. Hence, the service providers are flexible in allocating and de-allocating computing resources among the co-located applications.

Therefore, the service providers are also responsible for ensuring to meet their customer's application SLOs. For example, an enterprise can have the following application SLA with a service provider for one of its application.

LIFE CYCLE OF SLA:

Each SLA goes through a sequence of steps starting from identification of terms and conditions, activation and monitoring of the stated terms and conditions, and eventual termination of contract once the hosting relationship ceases to exist. Such a sequence of steps is called SLA life cycle and consists of the following five phases:

1. Contract definition
2. Publishing and discovery
3. Negotiation
4. Operationalization
5. De-commissioning

Here, we explain in detail each of these phases of SLA life cycle.

Contract Definition. Generally, service providers define a set of service offerings and corresponding SLAs using standard templates. These service offerings form a catalog. Individual SLAs for enterprises can be derived by customizing these base SLA templates.

Publication and Discovery. Service provider advertises these base service offerings through standard publication media, and the customers should be able to locate the service provider by searching the catalog. The customers can search different competitive offerings and shortlist a few that fulfill their requirements for further negotiation.

Negotiation. Once the customer has discovered a service provider who can meet their application hosting need, the SLA terms and conditions needs to be mutually agreed upon before signing the agreement for hosting the application. For a standard packaged application which is offered as service, this phase could be automated. For customized applications that are hosted on cloud platforms, this phase is manual. The service provider needs to analyze the application's behavior with respect to scalability and performance before agreeing on the specification of SLA. At the end of this phase, the SLA is mutually agreed by both customer and provider and is eventually signed off. SLA negotiation can utilize the WS-negotiation specification [8].

Operationalization. SLA operation consists of SLA monitoring, SLA accounting, and SLA enforcement. SLA monitoring involves measuring parameter values and calculating the metrics defined as a part of SLA and determining the deviations. On identifying the deviations, the concerned parties are notified. SLA accounting involves capturing and archiving the SLA adherence for compliance.
As part of accounting, the application's actual performance and the performance guaranteed as a part of SLA is reported. Apart from the frequency and the duration of the SLA breach, it should also provide the penalties paid for each SLA violation. SLA enforcement involves taking appropriate action when the runtime monitoring detects a SLA violation. Such actions could be notifying the concerned parties, charging the penalties besides other things. The different policies can be expressed using a subset of the Common Information Model (CIM) [9]. The CIM model is an open standard that allows expressing managed elements of data center via relationships and common objects.

De-commissioning. SLA decommissioning involves termination of all activities performed under a particular SLA when the hosting relationship between the service provider and the service consumer has ended. SLA specifies the terms and conditions of contract termination and specifies situations under which the relationship between a service provider and a service consumer can be considered to be legally ended.

SLA MANAGEMENT IN CLOUD:

SLA management of applications hosted on cloud platforms involves five phases.

1. Feasibility
2. On-boarding
3. Pre-production
4. Production
5. Termination

Different activities performed under each of these phases are shown in Figure 16.7.

These activities are explained in detail in the following subsections.

Feasibility Analysis
MSP conducts the feasibility study of hosting an application on their cloud platforms. This study involves three kinds of feasibility: (1) technical feasibility, (2) infrastructure feasibility, and (3) financial feasibility. The technical feasibility of an application implies determining the following:

1. Ability of an application to scale out.
2. Compatibility of the application with the cloud platform being used within the MSP's data center.
3. The need and availability of a specific hardware and software required for hosting and running of the application.
4. Preliminary information about the application performance and whether they can be met by the MSP.

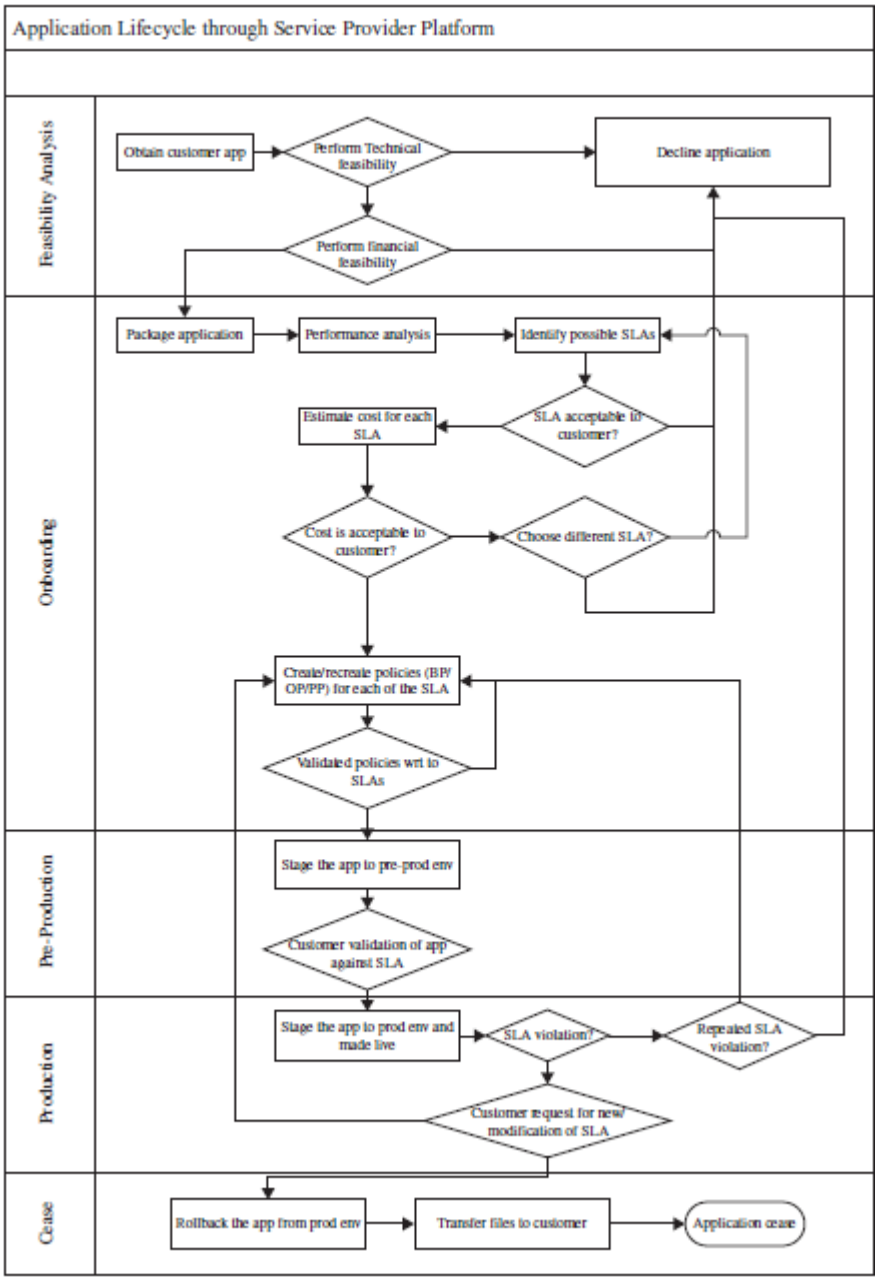


FIGURE 16.7. Flowchart of the SLA management in cloud.

On-Boarding of Application

Once the customer and the MSP agree in principle to host the application based on the findings of the feasibility study, the application is moved from the customer servers to the hosting platform. Moving an application to the MSP's hosting platform is called on-boarding [10]. As part of the on-boarding activity, the MSP understands the application runtime characteristics using runtime profilers. This helps the MSP to identify the possible SLAs that can be offered to the customer for that application. This also helps in creation of the necessary policies (also called rule sets) required to guarantee the SLOs mentioned in the application SLA. The application is accessible to its end users only after the on-boarding activity is completed.

Preproduction

Once the determination of policies is completed as discussed in previous phase, the application is hosted in a simulated production environment. It facilitates the customer to verify and validate the MSP's findings on application's runtime characteristics and agree on the defined SLA. Once both parties agree on the cost and the terms and conditions of the SLA, the customer sign-off is obtained. On successful completion of this phase the MSP allows the application to go on-live.

Production

In this phase, the application is made accessible to its end users under the agreed SLA. However, there could be situations when the managed application tends to behave differently in a production environment compared to the preproduction environment. This in turn may cause sustained breach of the terms and conditions mentioned in the SLA. Additionally, customer may request the MSP for inclusion of new terms and conditions in the SLA. If the application SLA is breached frequently or if the customer requests for a new non-agreed SLA, the on-boarding process is performed again. In the case of the former, on-boarding activity is repeated to analyze the application and its policies with respect to SLA fu lfillment. In case of the latter, a new set of policies are formulated to meet the fresh terms and conditions of the SLA.

Termination

When the customer wishes to withdraw the hosted application and does not wish to continue to avail the services of the MSP for managing the hosting of its application, the termination activity is initiated. On initiation of termination, all data related to the application are transferred to the customer and only the essential information is retained for legal compliance. This ends the hosting relationship between the two parties for that application, and the customer sign-off is obtained.