CC Notes

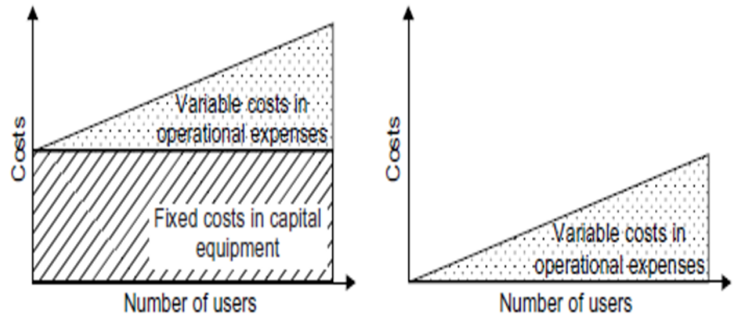
MOD-1

Essential Characteristics:

1. On-demand self-service
2. Broad network access
3. Resource pooling
4. Measured Service
5. Rapid elasticity

Cost Model:

1. Traditional Cost Model
2. Cloud Computing Cost Model



(a) Traditional IT cost model (b) Cloud computing cost model

Cloud Design Objectives:

1. Shifting computing from desktops to data centres
2. Service provisioning and cloud economics
3. Scalability in performance
4. Data privacy protection
5. High quality of cloud services
6. New standards and interfaces

Advantages:

1. Lower computer costs
2. Improved performance
3. Reduced software costs
4. Instant software updates
5. Improved document format compatibility
6. Unlimited storage capacity
7. Increased data reliability
8. Universal information access
9. Latest version availability
10. Easier group collaboration

11. Device independence

Disadvantages:

1. Requires a constant internet connection
2. Does not work well with low-speed connections
3. Features might be limited
4. Can be slow
5. Stored data might not be secured
6. Stored data can be lost
7. HPC Systems
8. may not be possible to run applications between cloud-based systems

Service Models:

1. **SaaS** – Google docs

   Used where:

   i) significant interplay between organization and outside world (email)
   ii) need for web or mobile access
   iii) only to be used for a short-term need
   iv) demand spikes significantly

   Not Used where:

   i) extremely fast processing of real time data is needed
   ii) legislation or other regulation does not permit data being hosted externally
   iii) existing on-premise solution fulfils all of the organization’s needs

2. **PaaS** – Ms Azure, Google App Engine, Heroku

   Used where:

   i) multiple developers will be working on a development project
   ii) other external parties need to interact with the development process
   iii) to automate testing and deployment services

iv) to implement agile software development methodologies

<u>Not Used where:</u>

i) proprietary languages or approaches would impact on the development process
ii) proprietary language would hinder later moves to another provider – concerns are raised about vendor lock in
iii) application performance requires customization of the underlying hardware and software

3. **IaaS** – Amazon EC2, S3

   <u>Used where:</u>

   i) significant spikes and troughs in terms of demand on the infrastructure
   ii) new organizations without the capital to invest in hardware
   iii) the organization is growing rapidly and scaling hardware would be problematic
   iv) pressure on the organization to limit capital expenditure and to move to operating expenditure
   v) trial or temporary infrastructural needs

   <u>Not Used where:</u>

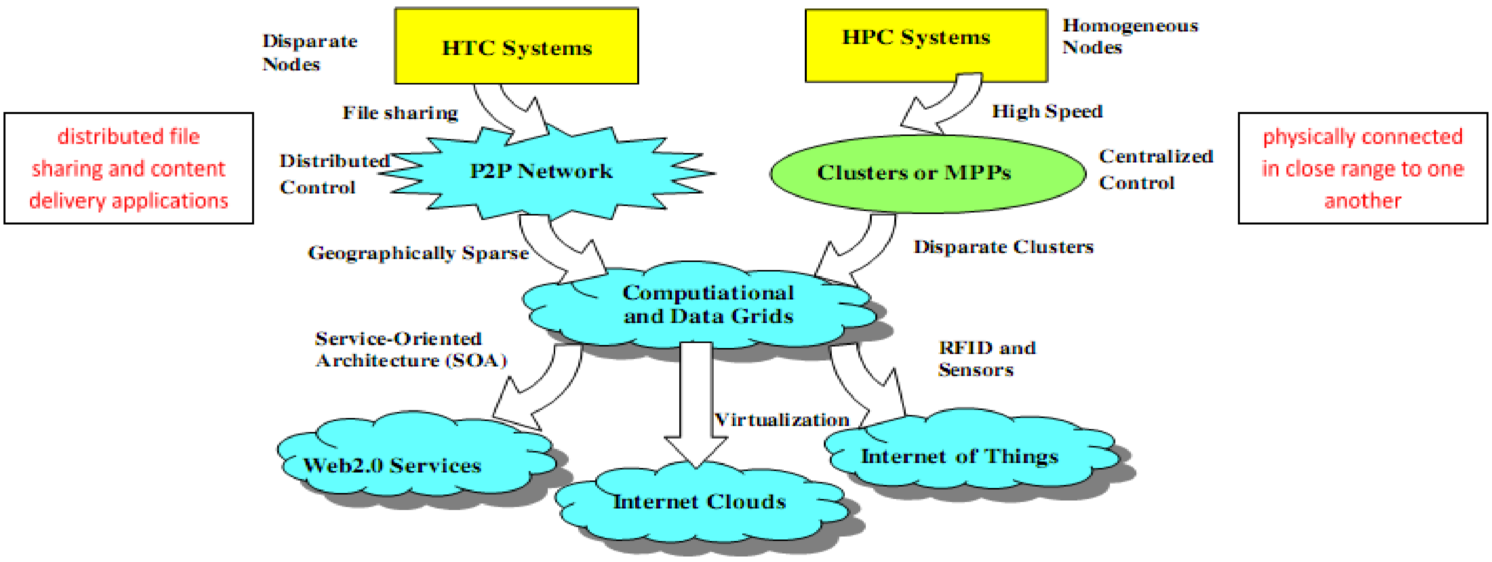
   i) regulatory compliance makes the offshoring or outsourcing of data storage and processing difficult
   ii) highest levels of performance are required
   iii) on-premise or dedicated hosted infrastructure has the capacity to meet the organization’s needs

<u>Deployment Models:</u>

1. **Public Cloud** (Google Doc, Spreadsheet, Google App Engine, Microsoft Windows Azure, IBM Smart Cloud, Amazon EC2)
2. **Private Cloud** - On-site and Outsourced (Window Server ‘Hyper-V’, Amazon VPC, VMware Cloud Infrastructure Suite)

3. **Hybrid Cloud** (Windows Azure, VMware vCloud, Cloud Bursting for load balancing between clouds {if an organisation using a private cloud reaches 100 percent of its resource capacity, the overflow traffic is directed to a public cloud so there is no interruption of services})
4. **Community Cloud** (Google Apps for Government, Microsoft Government Community Cloud)

<u>Evolutionary trend:</u>



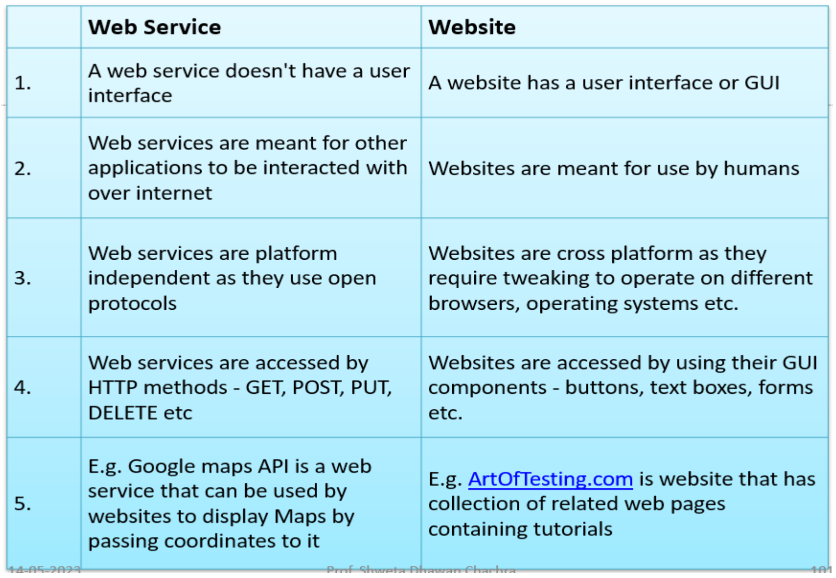
<u>Building cloud computing environments:</u>

1. Application development (webapps, data intensive or compute-intensive applications, scientific apps)
2. Infrastructure and system development (Distributed computing, Virtualization, Service orientation, Web 2.0/Web Services)

<u>Web Service</u>:

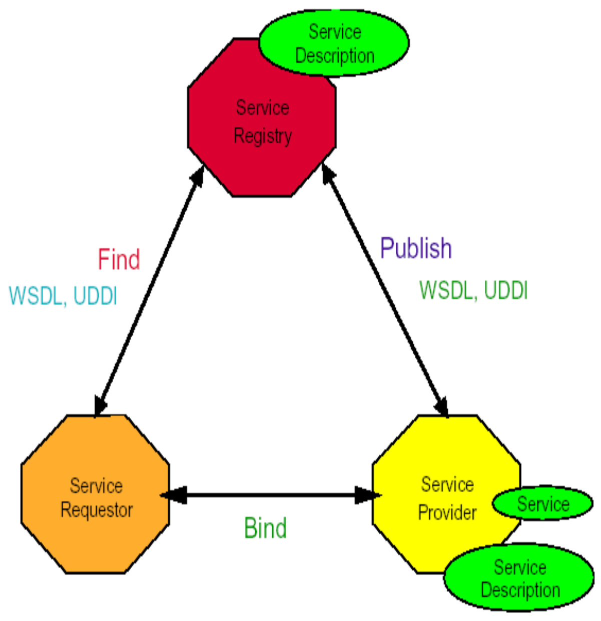
1. means by which computers talk to each other over the web using HTTP and other universally supported protocols
2. XML is used to encode all communications to a web service
3. not tied to any one OS or programming language
4. HTTP (Hypertext Transport Protocol)
5. SOAP (Simple Object Access Protocol) - standard way for communication
6. UDDI (Universal Description, Discovery and Integration)
7. WS-POLICY (Web Services Policy)

8. XML – eXtensible Markup Language
9. WSDL – Web Services Description Language



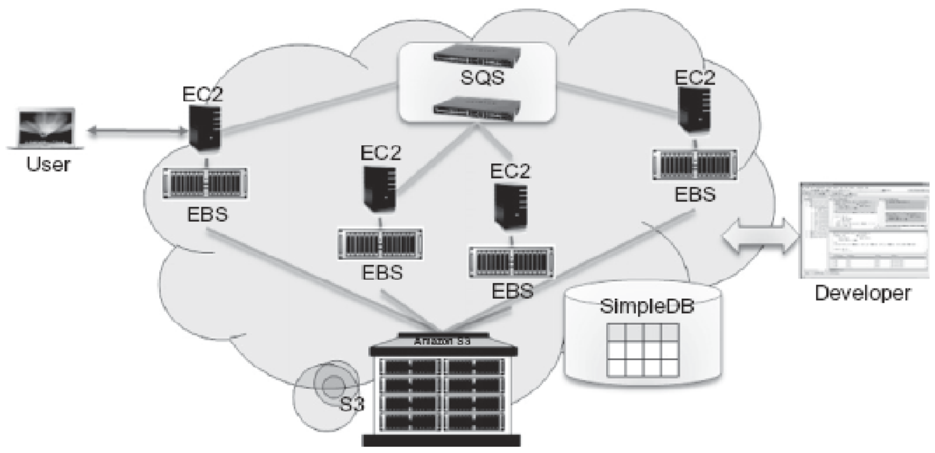
	Web Service	Website
1.	A web service doesn't have a user interface	A website has a user interface or GUI
2.	Web services are meant for other applications to be interacted with over internet	Websites are meant for use by humans
3.	Web services are platform independent as they use open protocols	Websites are cross platform as they require tweaking to operate on different browsers, operating systems etc.
4.	Web services are accessed by HTTP methods - GET, POST, PUT, DELETE etc	Websites are accessed by using their GUI components - buttons, text boxes, forms etc.
5.	E.g. Google maps API is a web service that can be used by websites to display Maps by passing coordinates to it	E.g. ArtOfTesting.com is website that has collection of related web pages containing tutorials

Service Oriented Architecture (SOA) by IBM:



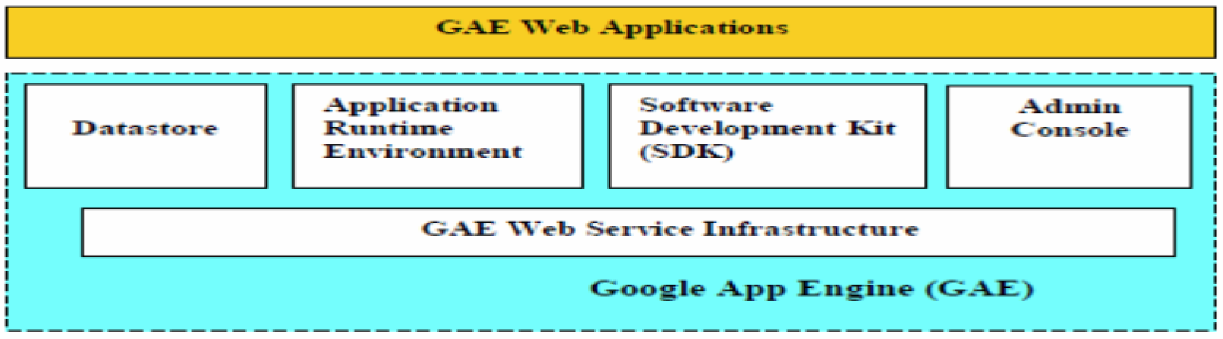
1. **Roles**: Service provider, Service requestor, Service registry
2. **Operations**: Publish, Find, Bind
3. **Steps**:
   i) Client queries registry to locate service.
   ii) Registry refers client to WSDL document.
   iii) Client accesses WSDL document.
   iv) WSDL provides data to interact with Web service.
   v) Client sends SOAP-message request.
   vi) Web service returns SOAP-message response.

Amazon web services (AWS) - IaaS:



1. **S3** (Simple Storage Service): object-oriented storage service for users, organized into buckets
2. **EBS** (Elastic Block Service): provides the block storage interface which can be used to support traditional applications
3. **EC2** (Elastic Compute Cloud): provides customizable virtual hardware that can be used as the base infrastructure for deploying computing systems on the cloud, has capability to save a specific running instance as an image
4. **SQS** (Simple Queue Service): ensure a reliable message service between two processes
5. **SOAP**: to access their objects with either browsers or other client programs which support the SOAP standard

Google App Engine (GAE) – PaaS (Python, Java, and Go):


GAE Web Applications
Datastore
Application Runtime Environment
Software Development Kit (SDK)
Admin Console
GAE Web Service Infrastructure
Google App Engine (GAE)

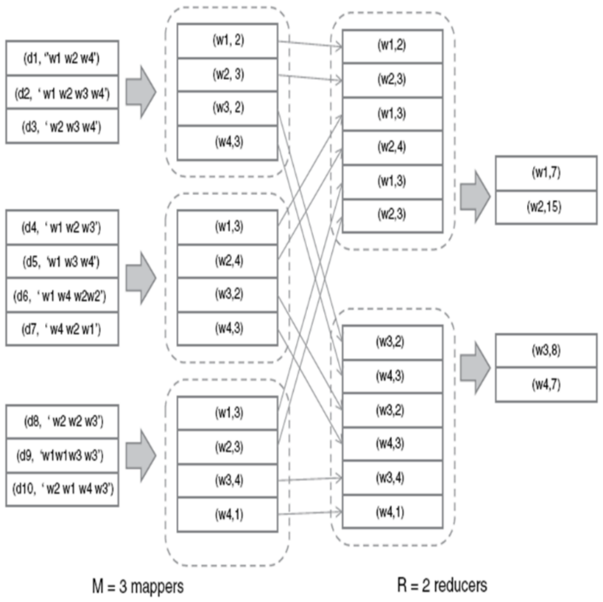
1. **Datastore** - object-oriented, distributed, structured data storage services based on BigTable techniques
2. **Application runtime environment**: platform for scalable web programming and execution (Python and Java)
3. **Software development kit (SDK)**: build and test applications on their own machines
4. After development, migrate application to AppEngine, set quotas to contain costs generated, and make application available to the world

Microsoft Azure:

1. platform for developing applications in the cloud
2. distributed applications
3. Roles: Web role (host a Web app), worker role (perform workload processing), virtual machine role (provides a virtual environment)

MapReduce (by Google):

1. Parallel and Distributed Programming Model for scalable data processing on large clusters over large data sets (web search)
2. 2 phases: Map operation & Reduce operation
3. coordinated by a single master process
4. Map function: $(d_k, [w_1 \ldots w_n]) \rightarrow [(w_i, c_i)]$.
5. Reduce function: $(w_i, [c_i]) \rightarrow \left(w_i, \sum_i c_i\right)$



Hadoop by Yahoo!:

1. open-source implementation of MapReduce
2. Software platform to write and run applications over the vast amounts of distributed data
3. easily scale Hadoop to store and process petabytes of data in the web space
4. only provide the input data and specify the map and reduce functions that need to be executed

Force.com – PaaS:

1. for social enterprise applications
2. create applications by composing ready-to-use blocks, set of components supporting all the activities
3. possible to develop your own components or integrate those available in AppExchange into your applications

SalesForce.com - SaaS:

1. Force.com platform is the basis for this
2. for customer relationship management

Manjrasoft Aneka – PaaS:

1. supports collection of programming abstractions for developing applications and a distributed runtime environment that can be deployed on heterogeneous hardware (clusters, networked desktop computers, and cloud resources)

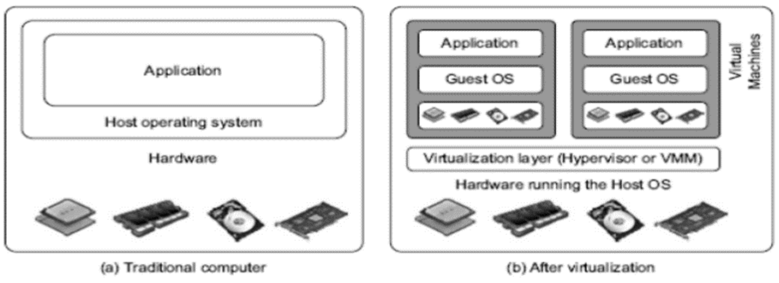
MOD-2:

Main Files in a VM:

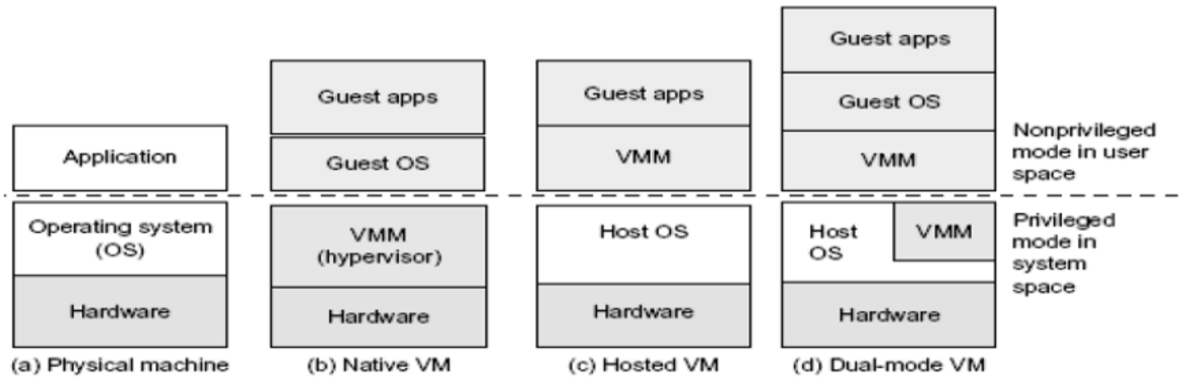
1. Configuration file
2. Virtual disk file
3. NVRAM settings file
4. Log file

Virtual Machines:

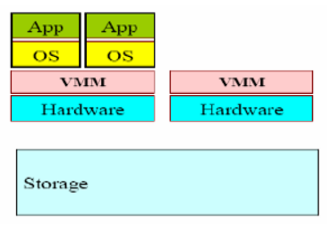
1. Resources are shared
2. Effective Utilization of Resources



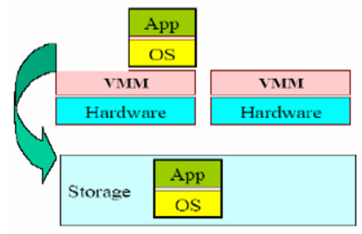
VM Architecture:



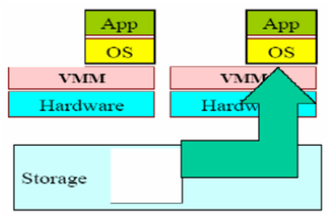
Primitive Operations in Virtual Machines:



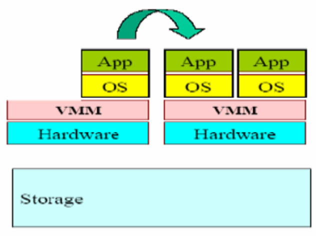
(a) Multiplexing



(b) Suspension (Storage)



(c) Provision (Resume)



(d) Life migration

Provisioning VM:

1. select a server along with appropriate OS
2. load appropriate software (device drivers, middleware and needed applications)
3. customize and configure M/c (IP address, Gateway) to configure associated n/w and storage resources

Live / Hot Migration - used for load balancing:

1. **Stage 0: Pre-Migration:** Active VM exists on physical host A. Alternate physical host may be selected for migration
2. **Stage 1: Reservation:** A request is issued to migrate an OS from host A to host B. A precondition is that the necessary resources exist on B and on a VM container of that size
3. **Stage 2: Iterative precopy:** During first iteration, all memory pages are transferred from A to B. Subsequent iterations, copy only those memory pages dirtied during the previous transfer phase
4. **Stage 3: Stop and Copy:** Running OS instance at A is suspended, its network traffic is redirected to B. CPU state and any remaining inconsistent memory pages are then transferred. At the end of stage, there is a consistent suspended copy of the VM at both A and B. The copy at A is considered primary. Resumed in case of failure
5. **Stage 4: Commitment:** Host B indicates to A that it has successfully received a consistent OS image. Host A acknowledges this message. Host A may now discard the original VM. Host B becomes the primary host
6. **Stage 5: Activation:** The migrated VM on B is now activated. connects to local devices. resumes normal operations.

Regular/Cold Migration:

1. no need for shared storage
2. no need for CPU compatibility checks
3. configuration files, including the NVRAM file (BIOS settings), log files, disks of the VM, are moved from source host to destination host's associated storage area
4. VM is registered with the new host
5. After migration, old version of VM is deleted from source host

Applications of Migration:

1. Load balancing: Move VMs to a less busy host, make use of newly-added capacity
2. Recovery from host failure: Restart VM on a different host
3. Maintenance: Planned Shutdown: Move VMs off a host before it is shut down

Levels of Virtualization:

1. instruction set architecture (ISA) level
2. hardware level
3. operating system level
4. library support level
5. application level

Instruction Set Architecture Level:

Transforming the physical architecture of the system's instruction set completely into software

1. Receiving a request to execute an application.
2. The application can include first application instructions from a guest instruction set architecture.
3. Loading an emulator
4. The emulator can translate the first application instructions into second application instructions from a host instruction set architecture.
5. Running the application by executing the second application instructions

Advantage:

1. Enables the host system to adjust to a change in the architecture of the guest system
2. infrastructure can be used for creating VMs on a platform
3. possible to run a large amount of legacy binary code written for various processors on any given new hardware host machine

Disadvantage:

1. interpreter interprets source instructions to target instructions one by one so poor performance

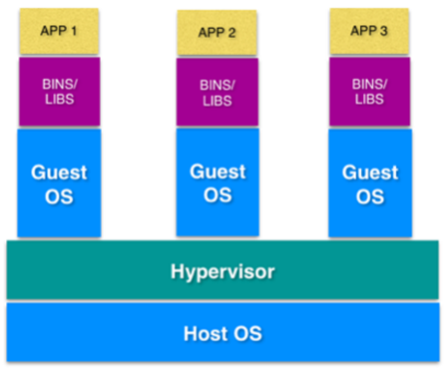
Eg: Qemu: machine emulator and virtualizer, Runs OS for any machine, on any supported architecture

Hardware Abstraction Level (Privileged Instructions):

1. virtualize Processors, memory, I/O devices etc
2. **Eg:** VMware, Virtual PC, Denali, Xen
3. **Privileged Instructions**: run only in Kernel Mode
4. **Advantages**: avoiding various risks and keeping individual virtual m/cs isolated
5. **HOW**: trapping the execution of privileged instructions by the virtual m/c, which must pass these instructions to the VMM for being handled properly
6. supports multiple OSs and applications to be run simultaneously
7. degree of isolation is high; implementation is less risky and maintenance is easy
8. **Disadvantages**: access to a raw computer, if physical and virtual OSs are the same then duplication of efforts, very expensive to implement (complexity)

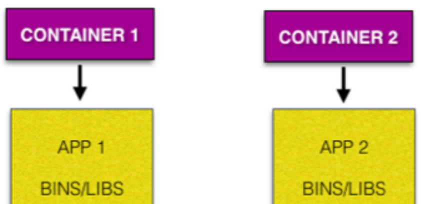
Operating System level:

1. abstraction layer between traditional OS and user applications
2. allocate hardware resources among a large number of mutually distrusting users
3. Advantages: overcome issues of redundancy (same physical and virtual OSs) and time consumption



VIRTUAL MACHINE ARCHITECTURE

Dockers container:



1. portable containers for software applications that has the application along with the associated dependency and configuration
2. container: environment that runs an application that is not dependent on the OS

Container:

1. kernel of the host OS
2. Each container runs isolated tasks
3. Advantages: no configuration entanglement, cannot harm the host machine nor come in conflict with other apps running in separate containers

Library Support Level:

1. Create execution environments for running alien programs on a platform rather than creating a VM to run the entire operating system.
2. It is done by API call interception and remapping
3. Advantage: low implementation effort
4. Disadvantages: poor application flexibility and isolation

Examples:

1. WINE (Wine Is Not an Emulator): support Windows applications on top of UNIX hosts (translates Windows API calls into POSIX calls on-the-fly)
2. vCUDA: allows applications executing within VMs to leverage GPU hardware acceleration

User-Application Level:

4. Virtualizes an application as a VM
5. High level language (HLL) VMs
6. Eg: Microsoft .NET CLR, Java Virtual Machine (JVM) {run Java programs as well as programs written in other languages but compiled to Java bytecode}
7. Advantage: has the best application isolation
8. Disadvantage: low performance, low application flexibility and high implementation complexity

LANDesk:

1. Fixing computer errors and updating systems from a remote location
2. Reports on installed software and hardware, allow remote assistance, and install operating system security patches

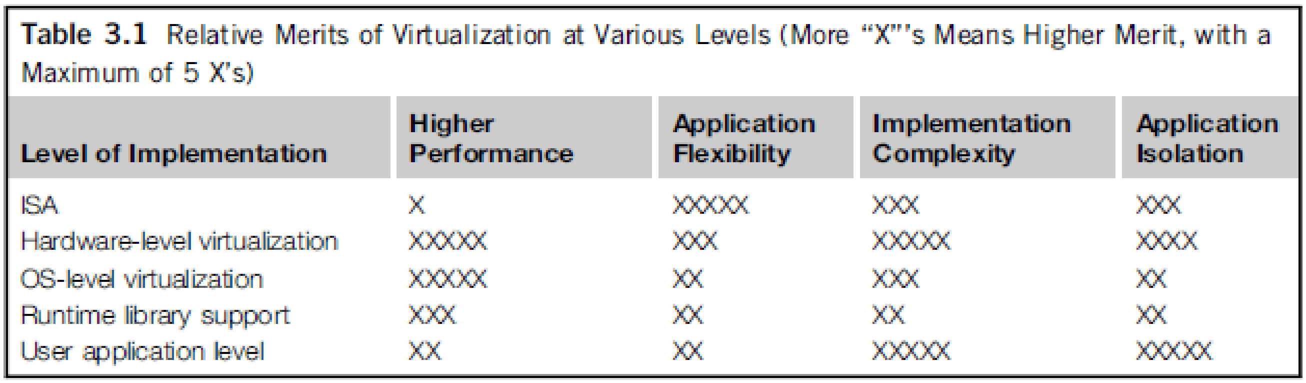


Table 3.1 Relative Merits of Virtualization at Various Levels (More "X"'s Means Higher Merit, with a Maximum of 5 X's)

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

Virtualization Design Requirements:

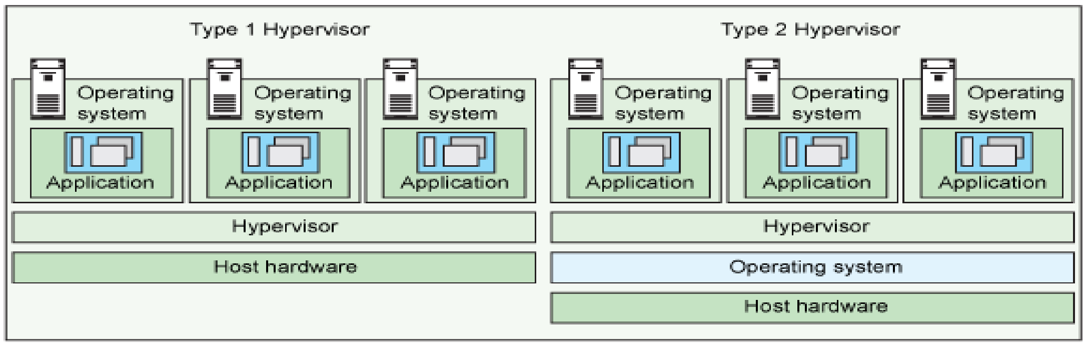
1. Equivalence Requirement - Needs to match the capabilities execute all the applications and programs of the physical system in its computational performance
2. Efficiency Requirement - as efficient in its performance as a real system
3. Resource Control requirement - resources (processors, memory, and I/O devices) must be managed and controlled effectively by the VMM

Virtualization Structures / Tools and Mechanisms:

1. Hypervisor and Xen Architecture
2. Binary Translation with Full Virtualization
3. Para-Virtualization with Compiler Support

Hypervisor / VMM:

hardware virtualization technique allowing multiple operating systems, called guests to run on a host machine

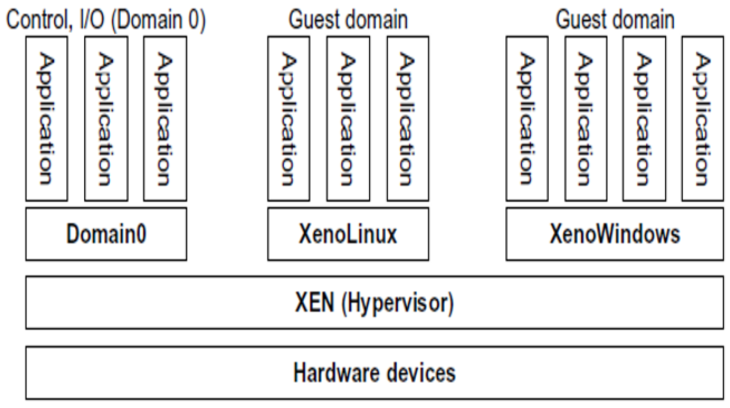


1. Type 1 Eg: Microsoft Hyper-V, Xen
2. Type 2 Eg: FreeBSD

Architectures:

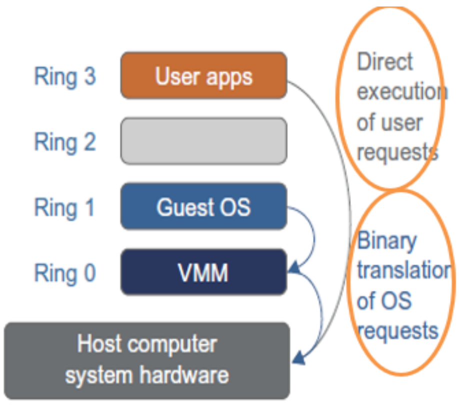
1. micro-kernel architecture - includes basic and unchanging functions like physical memory management and processor scheduling. The device drivers and other changeable components are outside the hypervisor (Microsoft Hyper-V, Xen)
2. monolithic hypervisor architecture - implements all the aforementioned functions, including those of the device drivers. Size of hypervisor code is larger (VMware ESX)

Xen Architecture:



1. Type 1 & microkernel Hypervisor
2. Primarily Based on Paravirtualization (Modifying portion of the guest operating system by Xen)
3. Recently been advanced to support full virtualization using hardware assisted virtualization
4. separates the policy (Domain 0) from the mechanism
5. core components - the hypervisor, kernel (HTTP), applications
6. Domain 0 - first loaded when Xen boots without any file system drivers being available, access hardware directly and manage devices, allocate and map hardware resources for the other guest domains (Domain U)
7. If Domain 0 is compromised, hacker can control entire system
8. HyperCalls - Sensitive system calls need to be re-implemented

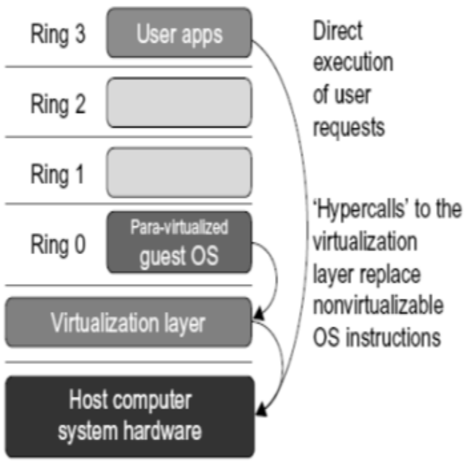
<u>Full Virtualization with Binary Translation:</u>


Ring 3
Ring 2
Ring 1
Ring 0
User apps
Guest OS
VMM
Host computer system hardware
Direct execution of user requests
Binary translation of OS requests

1. non-critical instructions run on the hardware directly
2. critical instructions are discovered and replaced with traps into the VMM to be emulated by software (by binary translation)
3. privileged, control- and behaviour-sensitive instructions trapped in VMM
4. Translation - offending instructions into an equivalent set of instructions that achieve the same goal without generating exceptions
5. Binary translation uses Caching
6. **Advantages**: No need to modify OS, Binary translation is only applied to a subset of the instruction set so reduces the impact on performance

7. **Disadvantages**: Translating instructions at runtime introduces additional overhead, time-consuming, the code cache increases the cost of memory usage
8. Eg: VMWARE

Para-Virtualization with Compiler Support:

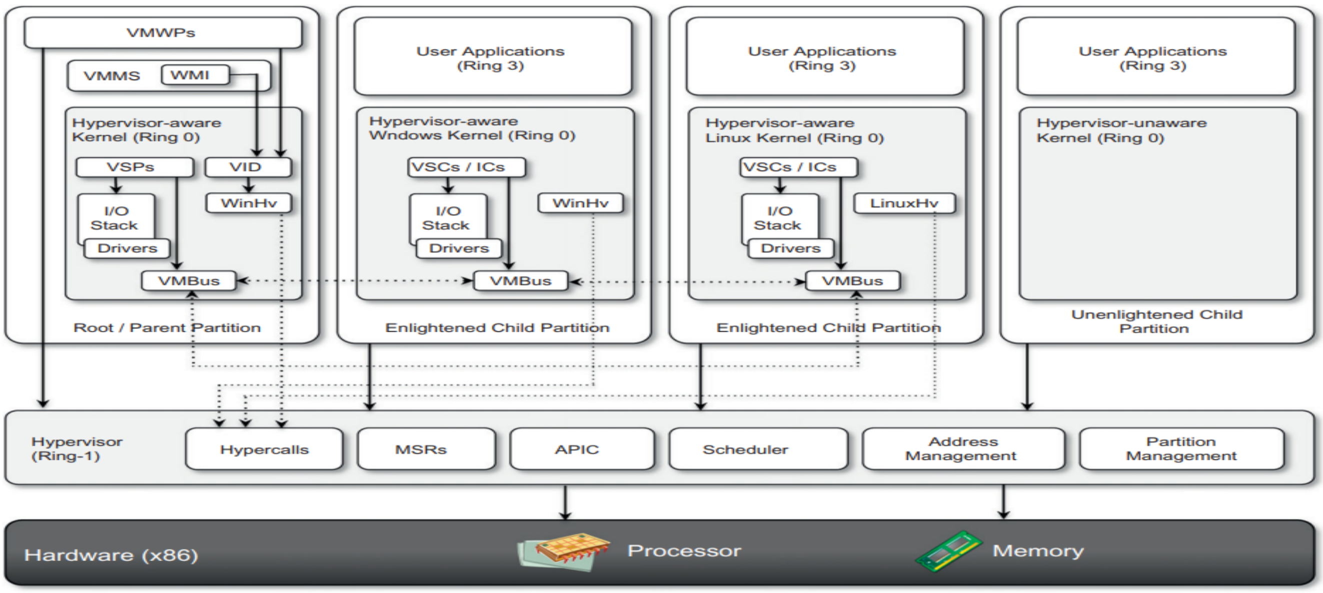

Ring 3
User apps
Ring 2
Ring 1
Ring 0
Para-virtualized guest OS
Virtualization layer
Host computer system hardware
Direct execution of user requests
'Hypercalls' to the virtualization layer replace nonvirtualizable OS instructions

1. needs to modify the guest OS so it can no longer run on the hardware directly
2. assisted by an intelligent compiler to replace the non-virtualizable OS instructions by hypercalls that communicate directly with the hypervisor or VMM
3. **Advantages:** reduces the virtualization overhead, and thus improve performance by modifying only the guest OS kernel

Full Virtualization vs. Para-Virtualization:

Full Virtualization	**Para-Virtualization**
does not need to modify guest OS	needs to modify guest OS
critical instructions are emulated by software through the use of binary translation	non-virtualizable instructions are replaced by hypercalls that communicate directly with the hypervisor or VMM
advantage is no need to modify OS	reduce the virtualization overhead, and improve performance
Disadvantage is approach of binary translation slows down the performance a lot	cost of maintaining Para virtualized OS is high

Microsoft Hyper-V:

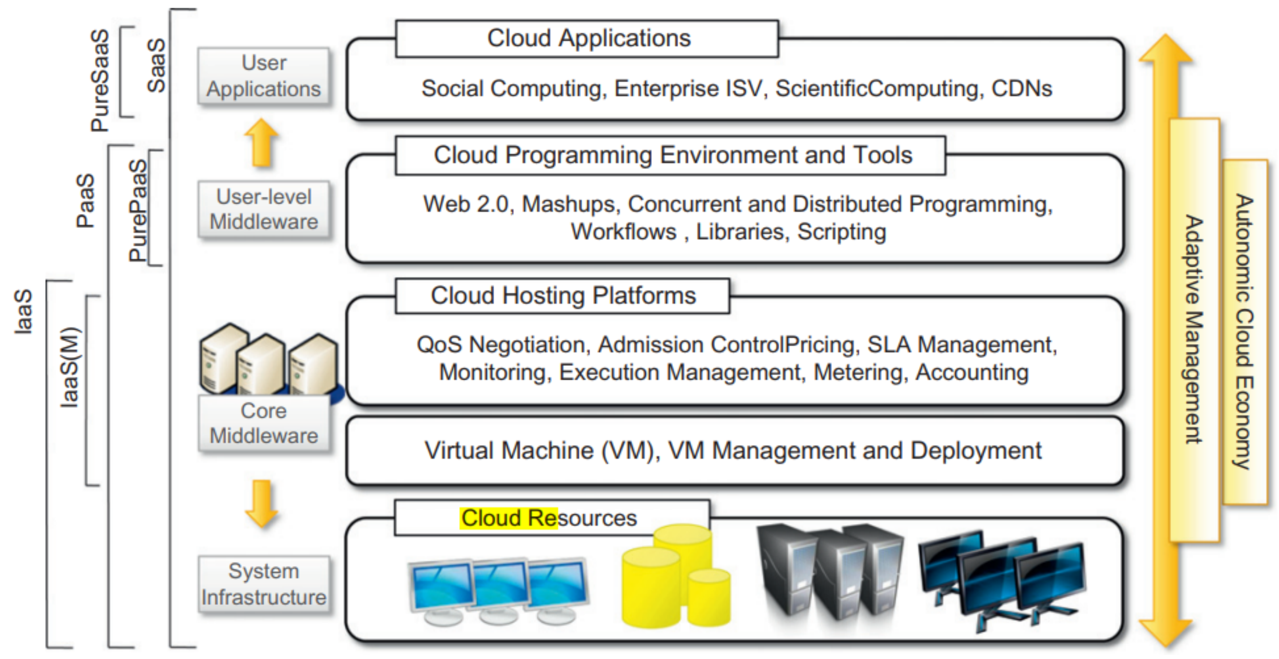

VMWPs
VMMS
WMI
Hypervisor-aware Kernel (Ring 0)
VSPs
VID
I/O Stack
Drivers
WinHv
VMBus
Root / Parent Partition
User Applications (Ring 3)
Hypervisor-aware Wndows Kernel (Ring 0)
VSCs / ICs
WinHv
VMBus
Enlightened Child Partition
Hypervisor-aware Linux Kernel (Ring 0)
LinuxHv
Hypervisor-unaware Kernel (Ring 0)
Unenlightened Child Partition
Hypervisor (Ring-1)
Hypercalls
MSRs
APIC
Scheduler
Address Management
Partition Management
Hardware (x86)
Processor
Memory

1. hypervisor-based approach to hardware virtualization
2. Type 1 Hypervisor
3. partition is a completely isolated environment in which an operating system is installed and run
4. Hyper-V takes control of the hardware, and the host operating system becomes a virtual machine instance with special privileges, called the parent partition
5. **Hypercalls interface** - used by drivers in the partitioned operating system to contact the hypervisor using the standard Windows calling convention
6. **Memory service routines (MSRs)** - control the memory and its access from partitions
7. **Scheduler** - schedules the virtual processors to run on available physical processors
8. **Address manager** - manage the virtual network addresses that are allocated to each guest OS
9. **Partition manager** - in charge of performing partition creation, finalization, destruction, enumeration, and configurations
10. **Advanced programmable interrupt controller (APIC)** - manages the signals coming from the underlying hardware when some event occurs (timer expired, I/O ready, exceptions and traps)

<u>VMware</u>:

supports End-user (desktop) virtualization, Server virtualization, Infrastructure virtualization and cloud computing solutions

The cloud reference model:


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

Pricing models:

1. Tiered pricing
2. Per-unit pricing
3. Subscription-based pricing

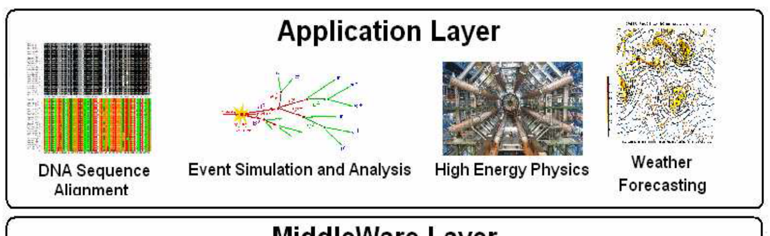
Open challenges:

1. Cloud definition
2. Cloud interoperability and standards
3. Scalability and fault tolerance
4. Security, trust, and privacy
5. Organizational aspects

MOD-5

Data-Centre Construction Requirements:

1. processor sockets
2. multicore CPU
3. internal cache hierarchy
4. local shared and coherent DRAM

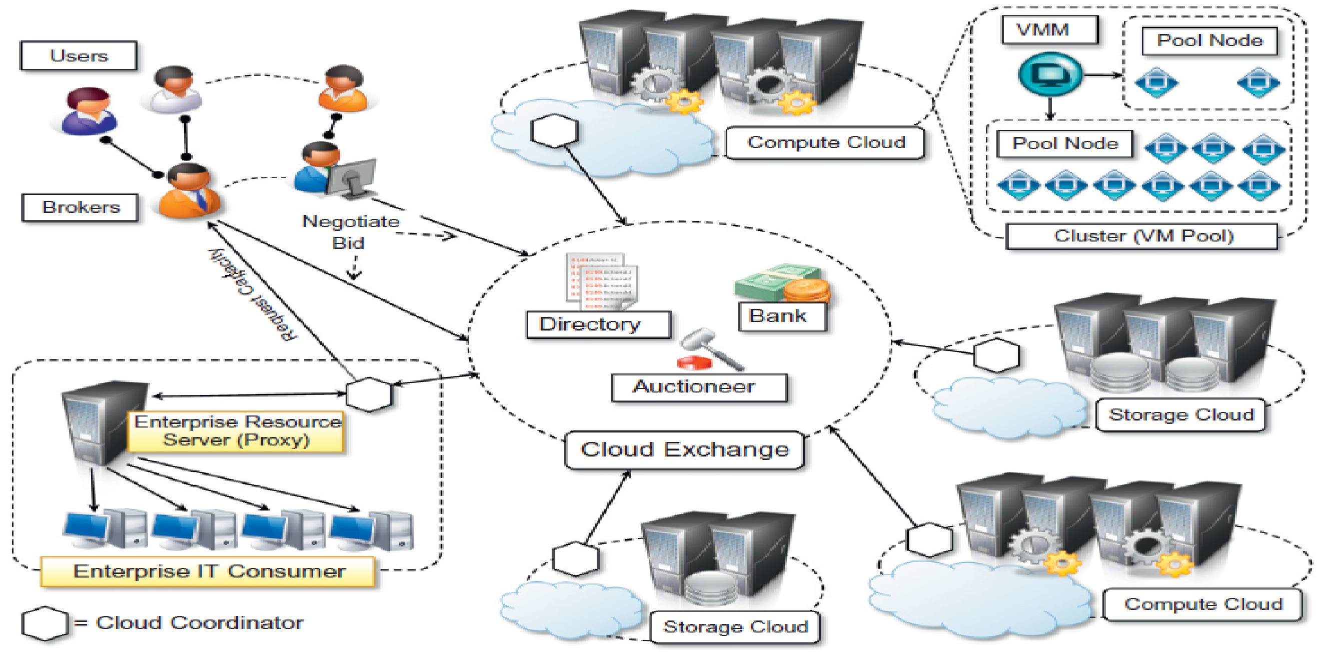


5. directly attached disk drives

Power management issues in distributed computing platforms:

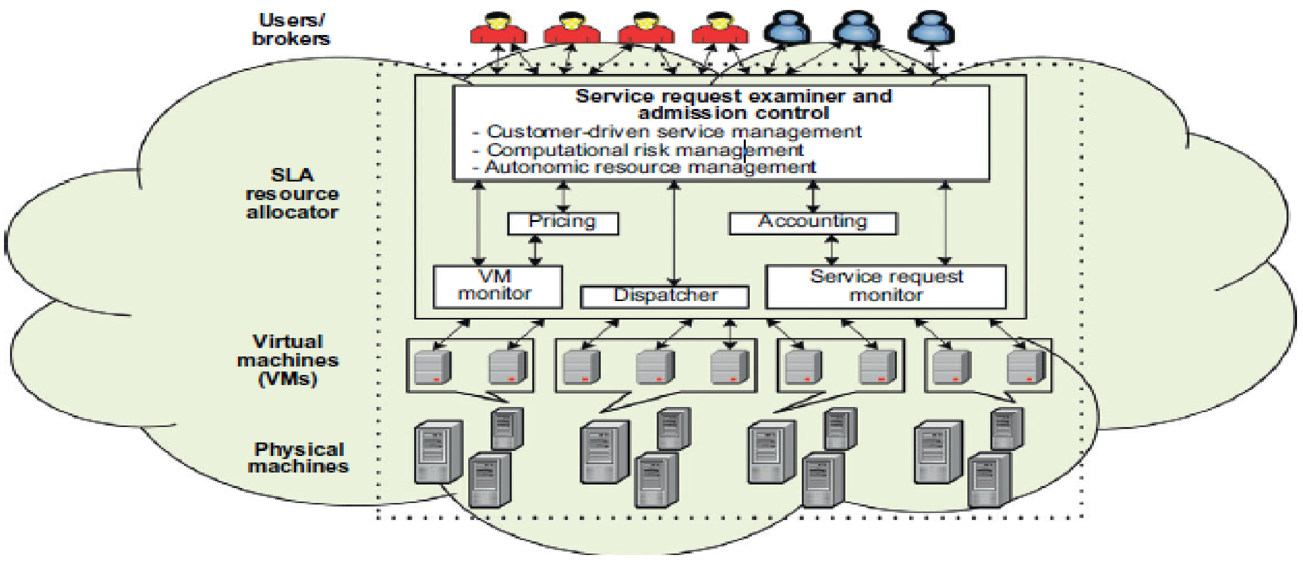
1. application layer
2. middleware layer
3. resource layer - Dynamic power management (DPM), Dynamic voltage-frequency scaling (DVFS)
4. network layer

Market-oriented cloud computing (MOCC):



Directory, Auctioneer, Bank, Cloud Exchange (CEx), Cloud Broker, Cloud Coordinators

Market-Oriented Cloud Architecture:


Users/ brokers
SLA resource allocator
Virtual machines (VMs)
Physical machines
Service request examiner and admission control
- Customer-driven service management
- Computational risk management
- Autonomic resource management
Pricing
Accounting
VM monitor
Dispatcher
Service request monitor

<u>Third Party Cloud Services:</u>

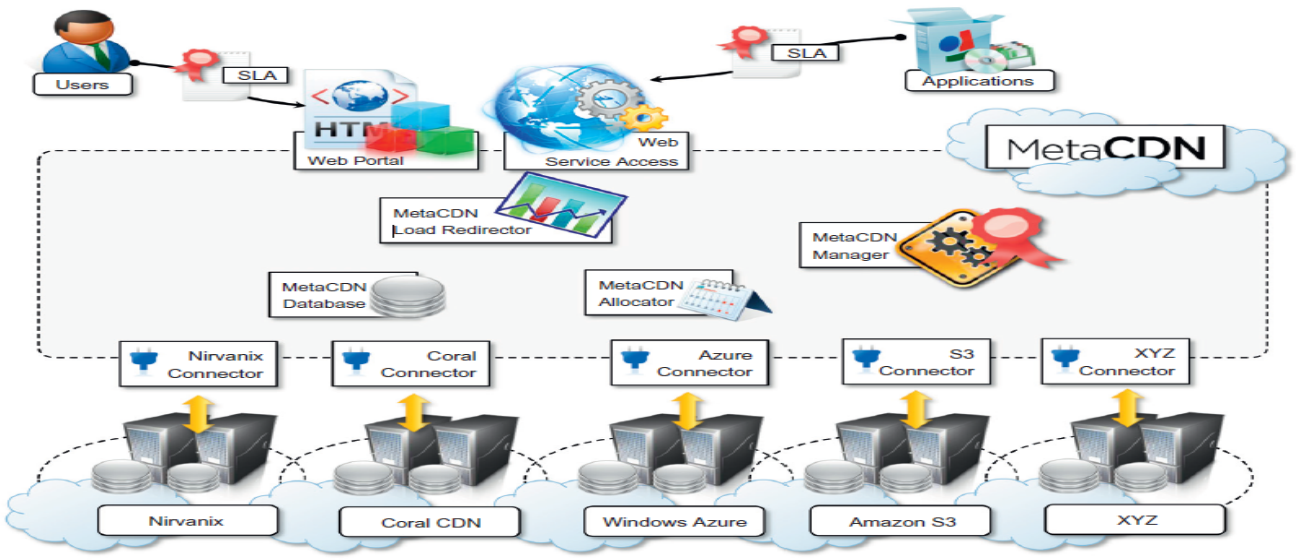
1. MetaCDN
2. SpotCloud

<u>CDN (content delivery / distribution network):</u>

1. network of interconnected servers that speeds up webpage loading for data-heavy applications
2. website content is stored on CDN servers geographically closer to the users
3. reduce the delay in communication created by a network's design
4. decrease web traffic to the web server, reduce bandwidth consumption, and improve the user experience of your applications
5. Eg: Akamai, Coral

<u>MetaCDN:</u>

1. coordinates the service offerings of different cloud storage vendors and uses them as distributed elastic storage on which the user content is stored
2. enables the uniform use of heterogeneous storage clouds as a single, large, distributed content delivery network
3. leverages Tier-1 Cloud Storage & CDN suppliers such as Amazon, Microsoft and Google to offer enterprise-class content delivery, video encoding and streaming services on an unmatched global scale
4. main operations - creation of deployments over storage clouds and their management



Deployment options:

1. Coverage and performance-optimized deployment - deploy as many replicas as possible to all available locations
2. Direct deployment - allows the selection of the deployment regions for the content and will match the selected regions with the supported providers
3. Cost-optimized deployment - deploys as many replicas in the locations identified by the deployment request and budget will be used to deploy the replicas and keep them active for as long as possible
4. QoS optimized deployment - selects the providers that can better match The QoS requirements attached to the deployment

Components:

1. MetaCDN Manager - ensuring that all the content deployments are meeting the expected QoS
2. MetaCDN QoS Monitor - monitors data transfers to assess the performance of each provider
3. Load Redirector - redirecting user content requests to the most suitable replica
4. connectors - manage Interactions with storage clouds

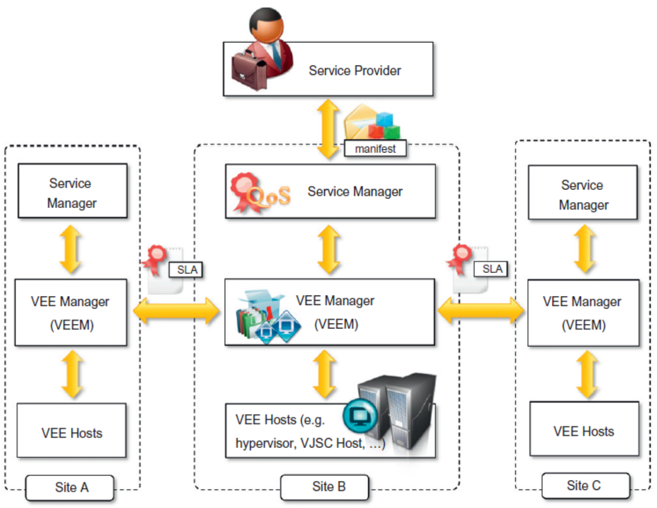
<u>SpotCloud:</u>

1. online portal that implements a virtual marketplace, where sellers and buyers can register and trade cloud computing services
2. market place operating in the IaaS sector
3. allows users with available computing capacity to easily turn themselves into service providers by deploying the runtime environment required by SpotCloud on their infrastructure

4. includes a complete bookkeeping of the transactions associated with the use of resources
5. provides vendor lock-in-free solution

Federated Clouds:

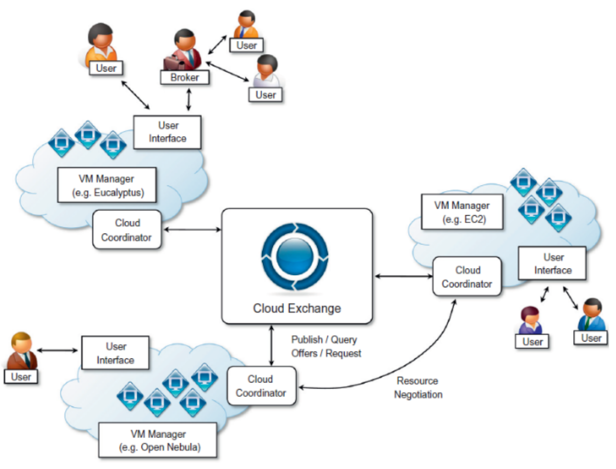
federation implies that there are agreements between the various cloud providers, allowing them to leverage each other's services in a privileged manner



Eg: **RESERVOIR** (Resources and Services Virtualization Without Barriers)

1. Supports providers of cloud infrastructures to dynamically partner with each other to extend their capabilities while preserving their administrative autonomy
2. Service Manager - Constitutes the front-end used by service providers to submit services. It deploys and provisions VEEs and monitors and enforces SLA compliance by controlling the capacity
3. Virtual Execution Environment (VEE) Manager - responsible for the optimal placement of VEEs into VEE hosts according to the constraints expressed by the Service Manager. Also interacts with VEE Managers in other sites to provision additional instances for the execution of service applications or move VEEs to other sites in case of overload
4. VEE Host (VEEH) - put into practice the IT management decisions regarding heterogeneous sets of virtualization platforms. Ensures networking among VEEs that belong to the same application. Encapsulates all platform-specific management through a standardized interface to the VEE Manager

Intercloud - Cloud of Clouds:



1. a composition of clouds that are interconnected by means of open standards to provide a universal environment that leverages cloud computing services
2. **CloudExchange** - offers services that allow providers to find each other & allowing parties to register and run auctions

3. **CloudCoordinator** - Front-end components (interact with the CloudExchange and with other coordinators) & Back-end components (allows the coordinator to learn about the current state of the datacentre to decide whether actions from the federation are required or not)

Monolithic Architecture:

1. all processes are tightly coupled and run as a single service
2. if one process of the application experiences a spike in demand, the entire architecture must be scaled
3. Adding or improving a monolithic application's features becomes more complex
4. many dependent and tightly coupled processes increase the impact of a single process failure
5. all teams in a monolith are interdependent

Microservices:

1. software is composed of small independent services that communicate over well-defined APIs
2. owned by small, self-contained teams
3. collection of services that are: Independently deployable, loosely coupled, organized around business capabilities, Owned by a small team
4. **Characteristics:** Autonomous & Specialized
5. **Eg**: Netflix, Spotify, eBay

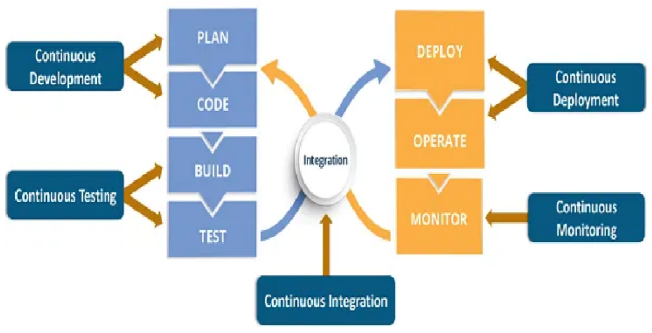
Benefits:

1. Agility
2. Flexible Scaling
3. Easy Deployment
4. Technological Freedom
5. Reusable Code
6. Resilience

DevOps:

1. continuous integration between deployment of code and testing

2. real-time monitoring and immediate feedback
3. Agile approach
4. DevOps Trinity (PPT): People and Culture, Processes and Practices, Tools and Technologies
5. Benefits: Continuous delivery of software, better collaboration between teams, Easy deployment, better efficiency and scalability, Errors are fixed at the initial stage, more security, less manual intervention (which means fewer chances of error)



6. Phases (8): Plan, Code, Build, Test, Deploy, Operate, Monitor, Integrate
7. 6 Cs of DevOps: Continuous Development, Continuous Integration, Continuous Testing, Continuous Deployment, Continuous Feedback, Continuous Monitoring