Cloud Computing

(MCCS-T-PE-021)

Dr. Debabrata Kar
Silicon Institute of Technology, BBSR

Disclaimer

The contents included in this PPT have been prepared by collecting information from various books, websites, Wikipedia, vendor websites, online tutorials, NPTEL videos and other sources. There is no content that is original and contributed by the author. Though utmost care has been taken to provide information that is correct & accurate, as the content is based on material collected from diverse sources, there may be unintentional ambiguity, incorrect or inaccurate information. The reader is advised to cross verify the content with other trusted sources and adopt what is correct.

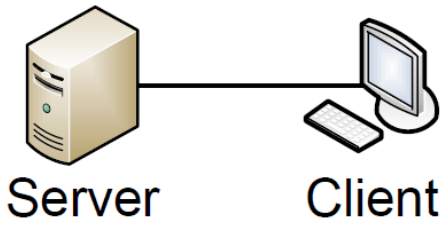
You are free to use this material at your own risk. The author shall not be responsible in any manner for any loss or damage caused by using the information from this PPT or transmitting them to other readers.

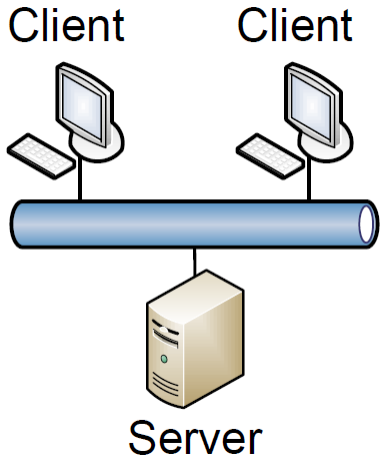
Books & References

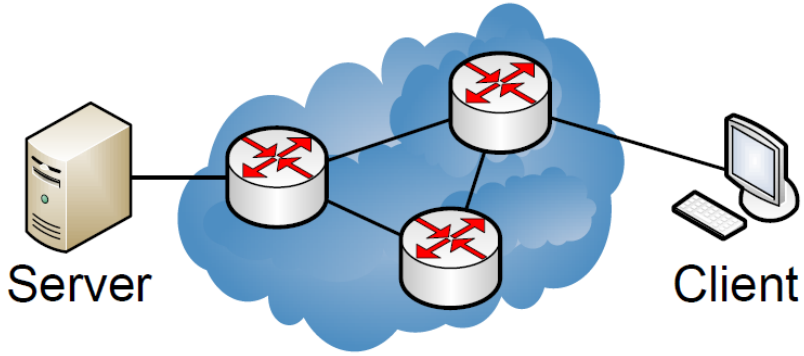
1. Cloud Computing – Bagha & Madisetti (UP)
2. Distributed & Cloud Computing
   - Hwang, Fox, Dongarra (Morgan Kaufman)
3. Cloud Computing Bible - Sosinsky (Willey)
4. Cloud Computing – A Practical Approach
   - Velte, Velte, Elsenpeter (Tata McGraw Hill)
5. Cloud Computing – Tomas Erl (PHI)
6. NPTEL Lectures & Videos
   - Prof. S. K. Ghosh, IIT Kharagpur
7. Various Internet Sources

Client-Server Systems

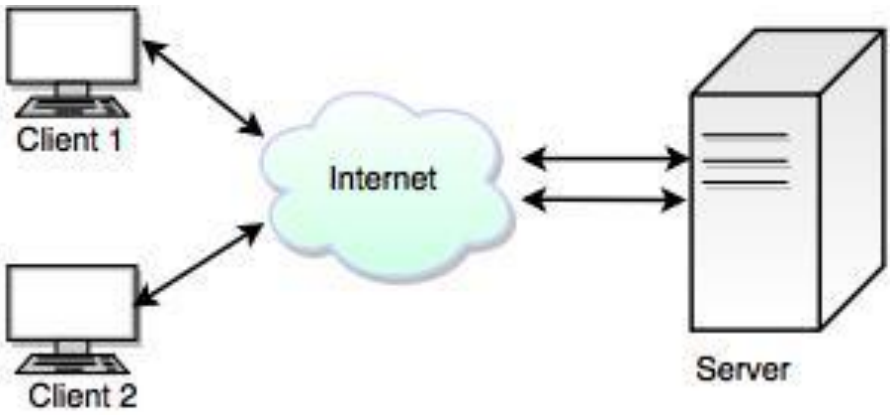
- One or more client computers depend on a Server computer to perform all computations.





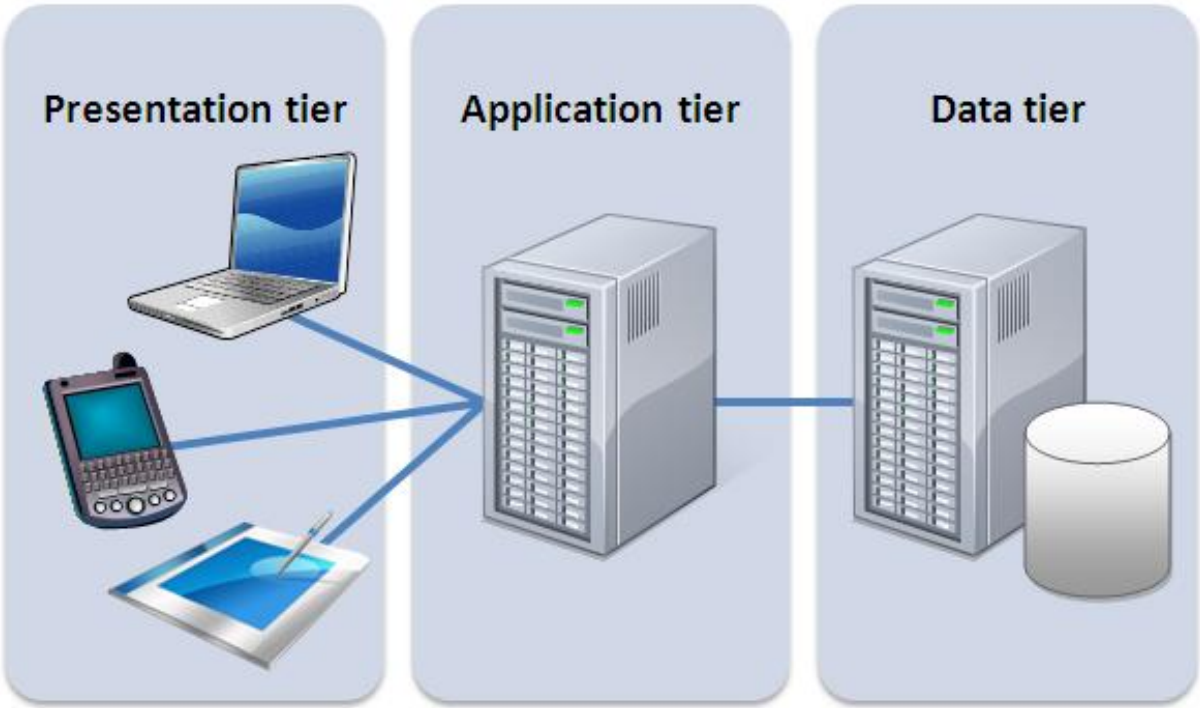


Client-Server Systems: 2-Tier & 3-Tier



← 2-Tier Architecture

3-Tier Architecture →



Thin & Fat Clients

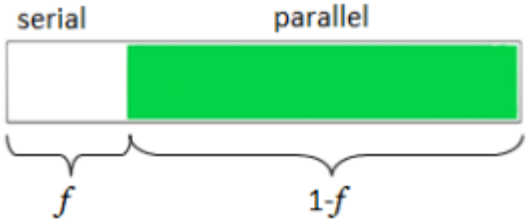
- In a Thin Client model, all of the application processing and data management is carried out on the server. The client is responsible only to run the presentation software.
- In a Fat Client model, the server is responsible for data management. The client implements the application logic and interacts with the system user. The fat client provides rich functionality *independent* of the server.

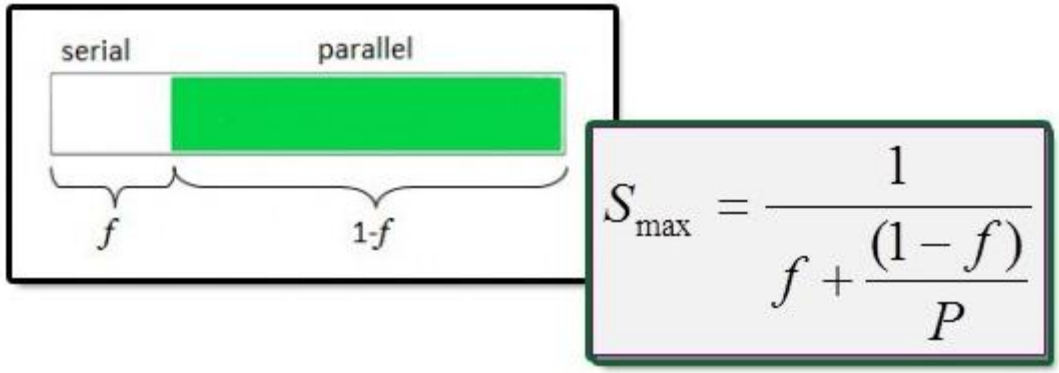
Centralized Computing

- Computing Resources:
  - Processors
  - Memory
  - Storage
  - Peripherals
- In Centralized Computing, ALL computing resources are centralized in one physical system. The resources are fully shared between all applications but are tightly coupled in one integrated Operating System.

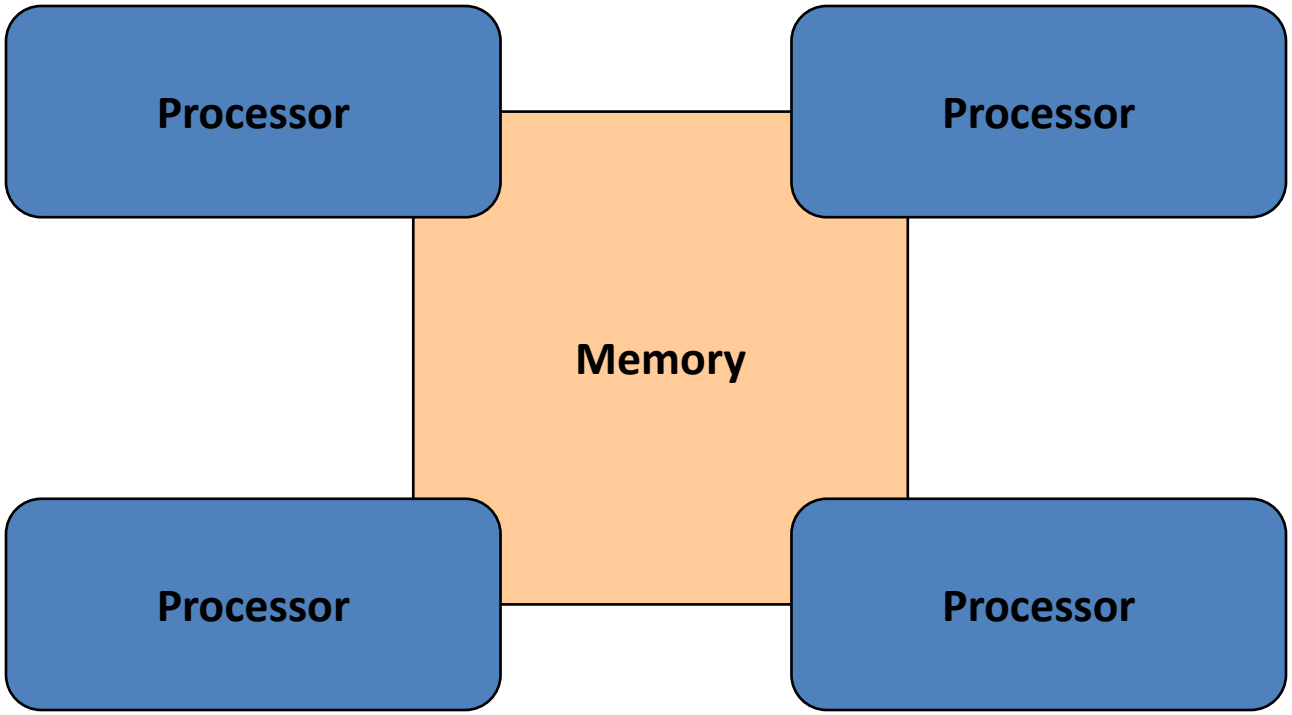
Centralized Computing

- Centralized computing is not sufficient for all computational needs → Parallel Computing
  - Approach: Divide & Conquer
  - Intention: Using ***P*** computers (or processors), we expect to complete the task ***P***-times faster.
  - Not Possible: **Amdahl's Law** states the limitation.



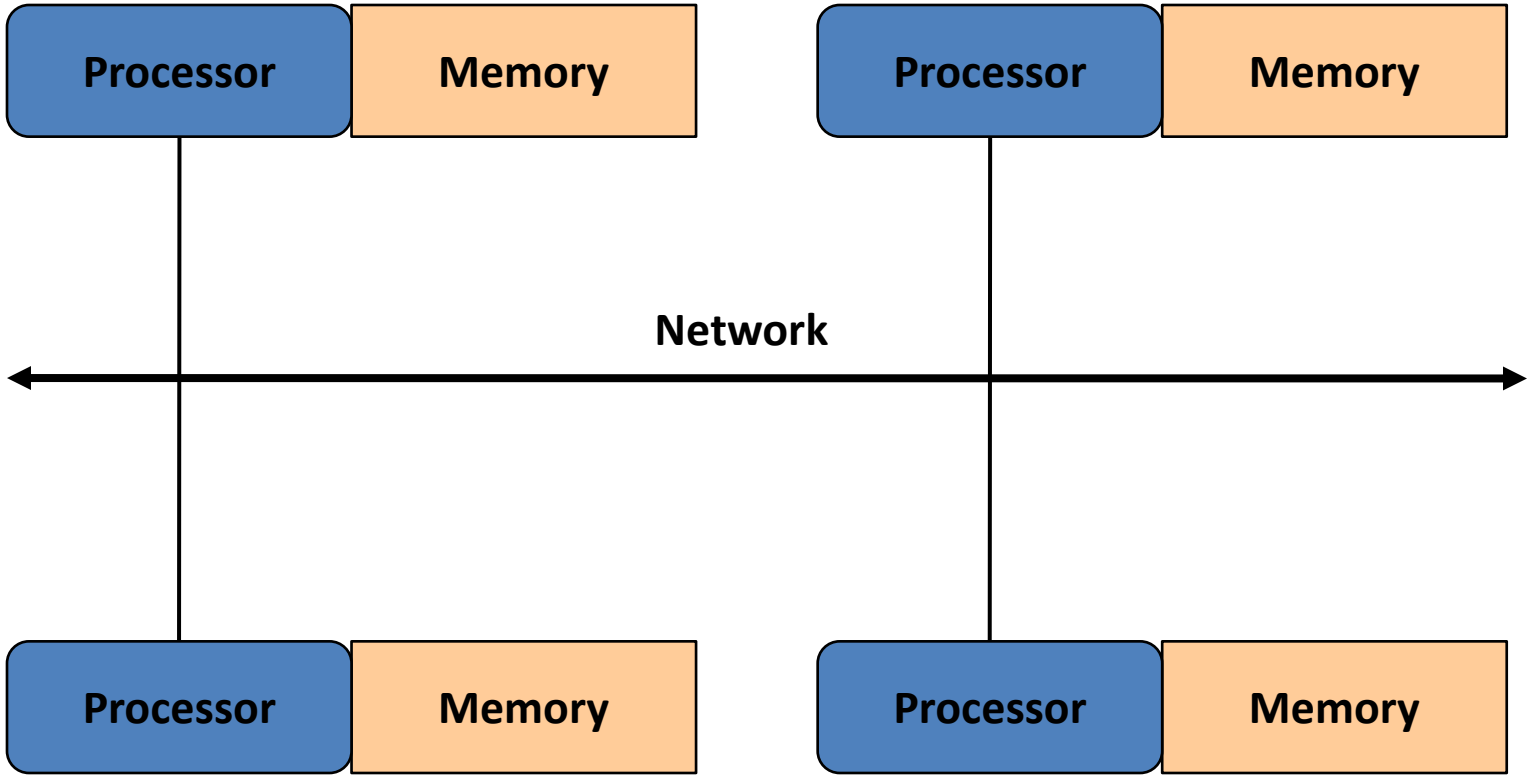


Parallel & Distributed Computing



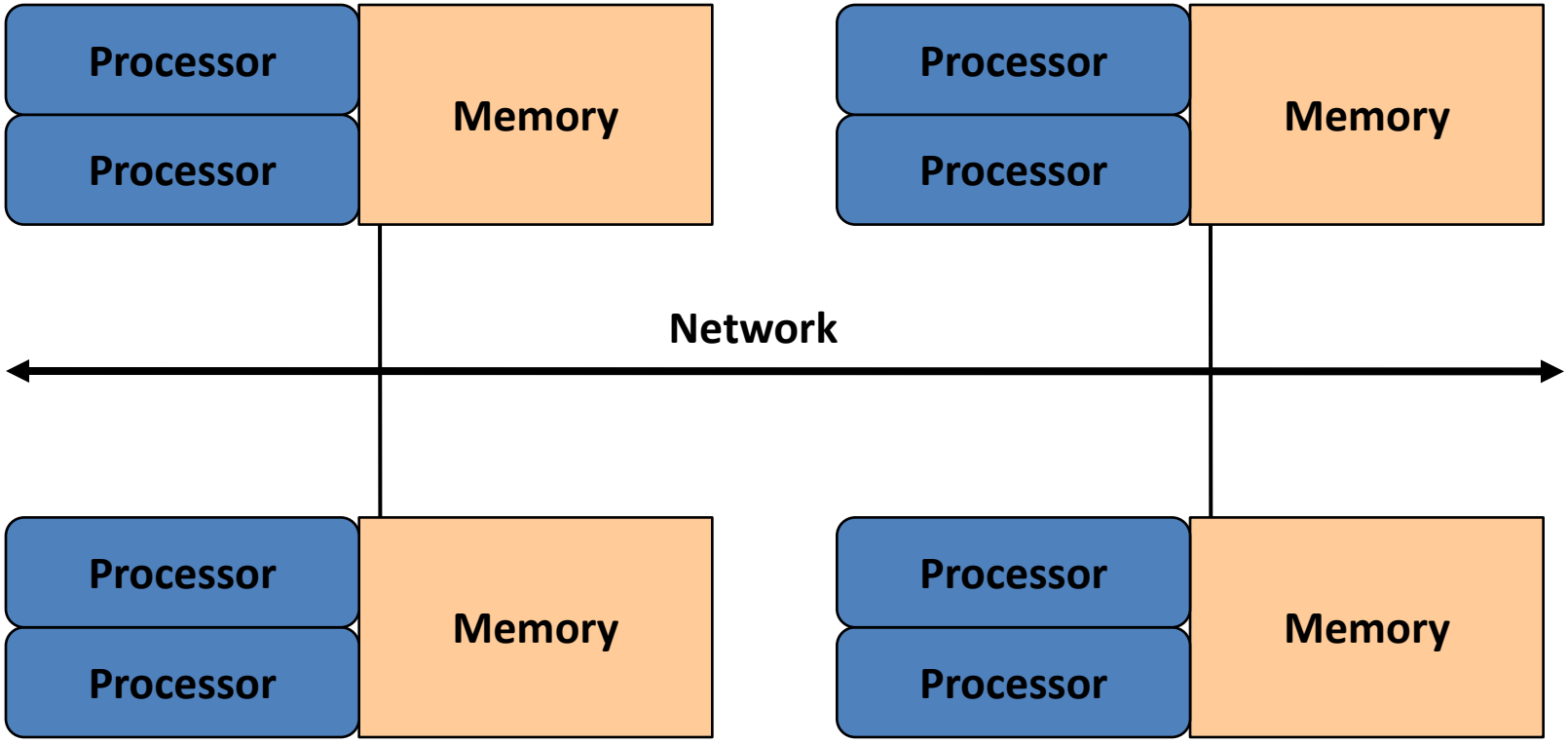
Multiple Processors with Centralized Shared Memory

Parallel & Distributed Computing



Multiple Processors with Distributed Memory

Parallel & Distributed Computing



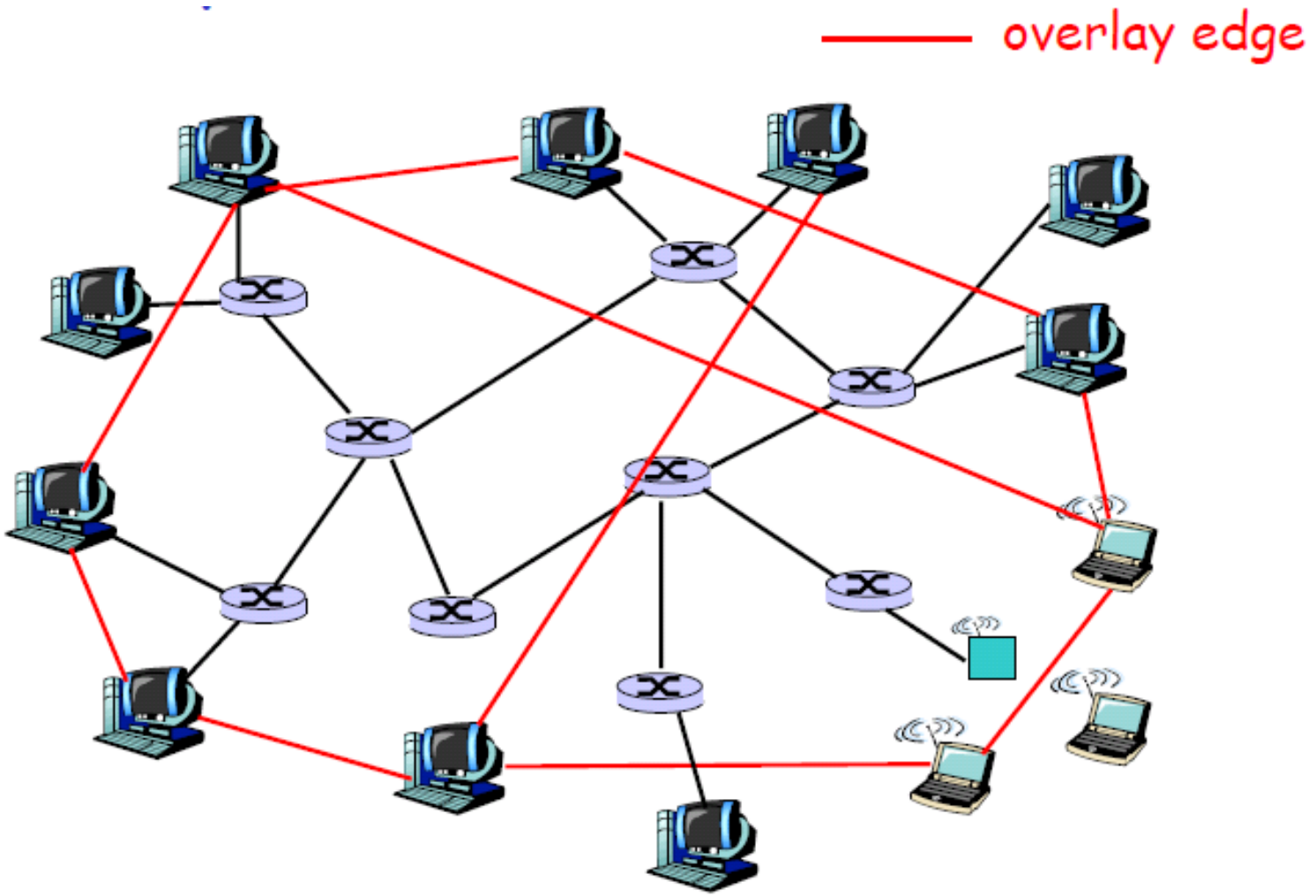
Multiple Processors with Hybrid Memory

Parallel vs. Distributed Computing

	Parallel Systems	Distributed Systems
Memory	**Tightly coupled shared** memory among all processors	**Distributed** memory Message passing, RPC, and/or use of distributed shared memory
Control	**Global clock** control SIMD, MIMD	**No global clock** control Synchronization algorithms needed
Processor interconnection	Order of **Tbps**	Order of **Gbps**
Main focus	**Performance** Scientific computing	Performance (**cost and scalability**) **- Reliability/Availability** **- Information/Resource Sharing**

Peer-to-Peer Systems

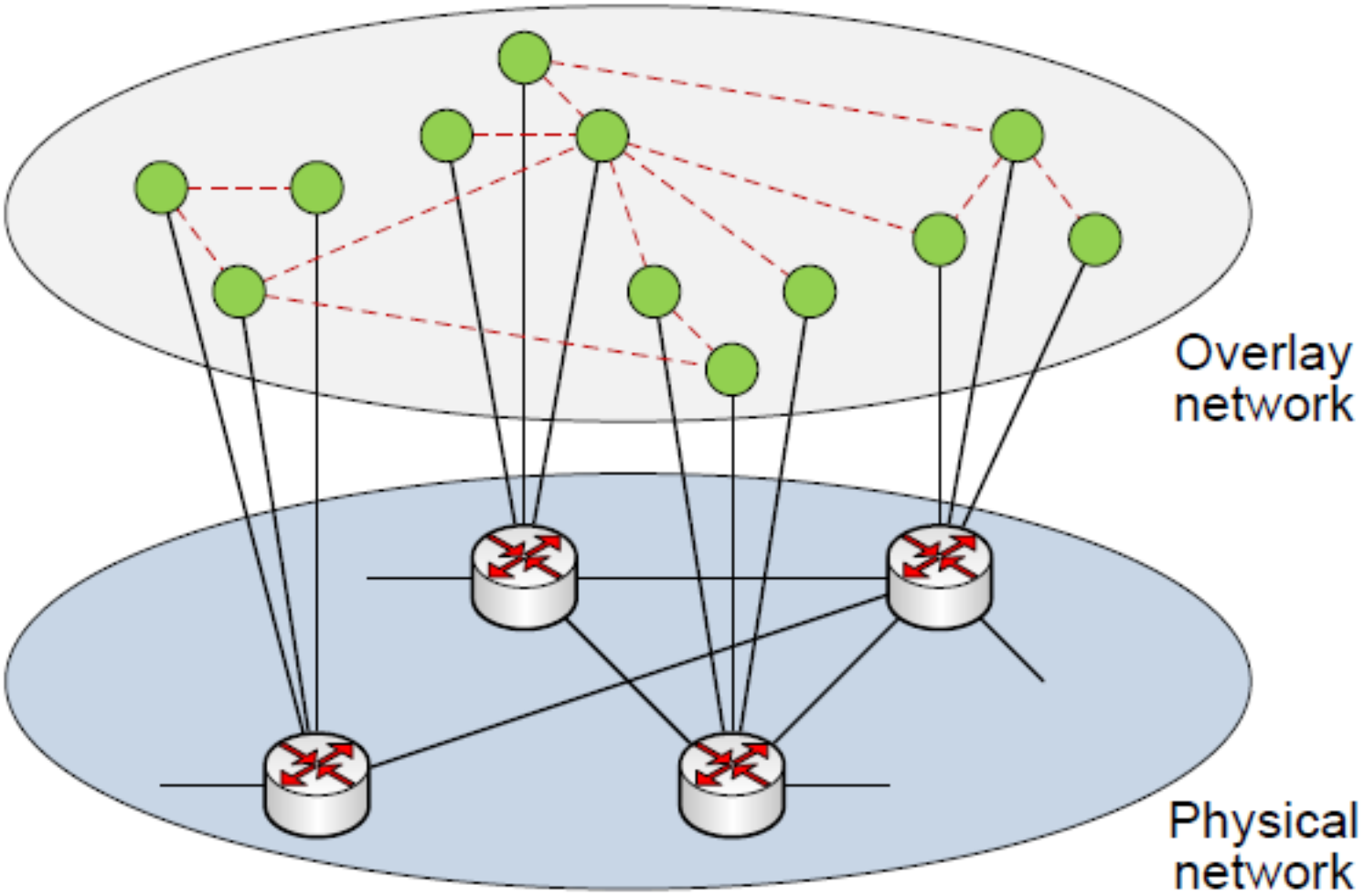
It is taken use by distributed computing.



A P2P network is an overlay network.

Each link between peers consists of one or more IP links.

Peer-to-Peer Systems



Peer-to-Peer Systems

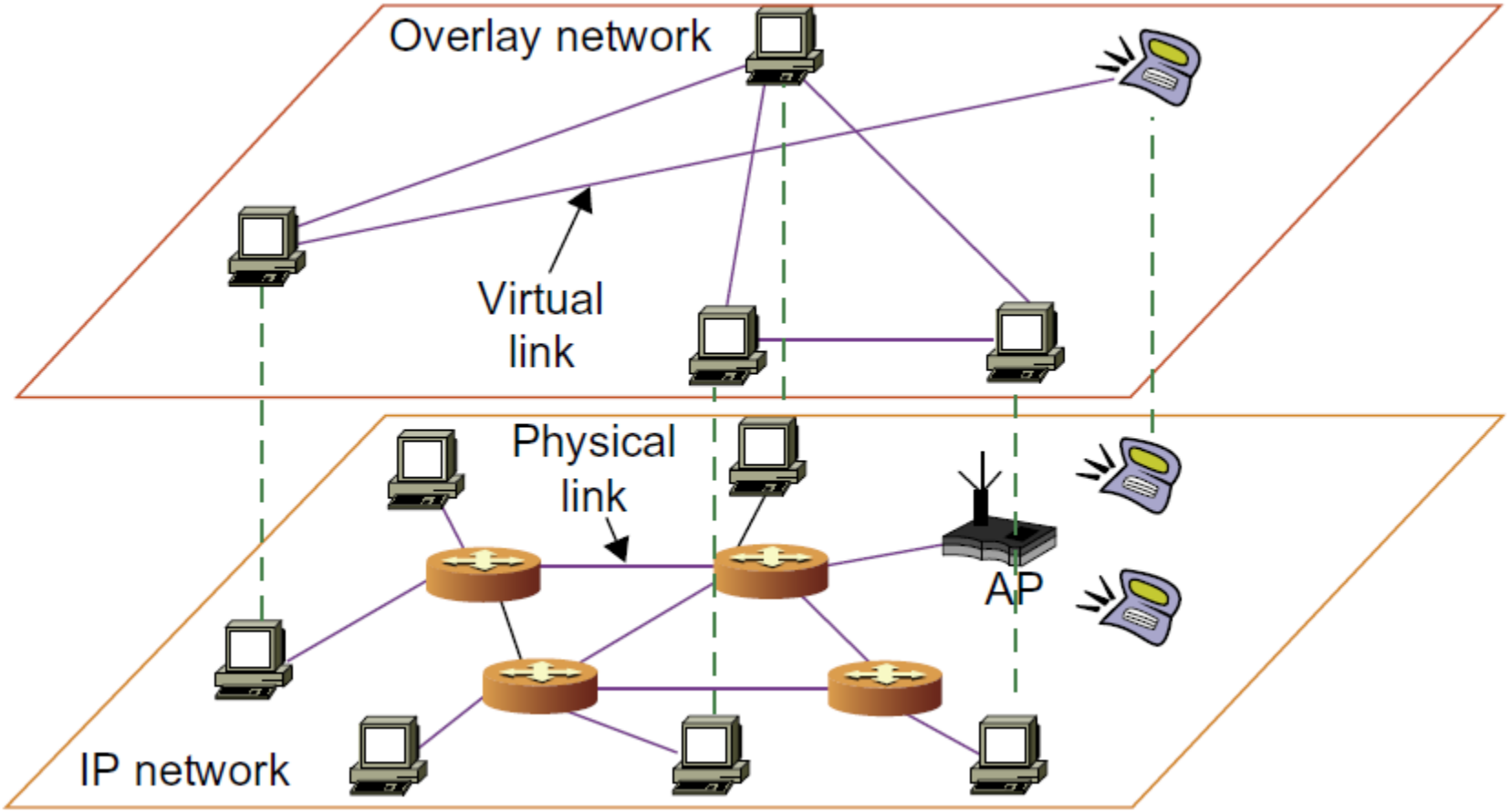
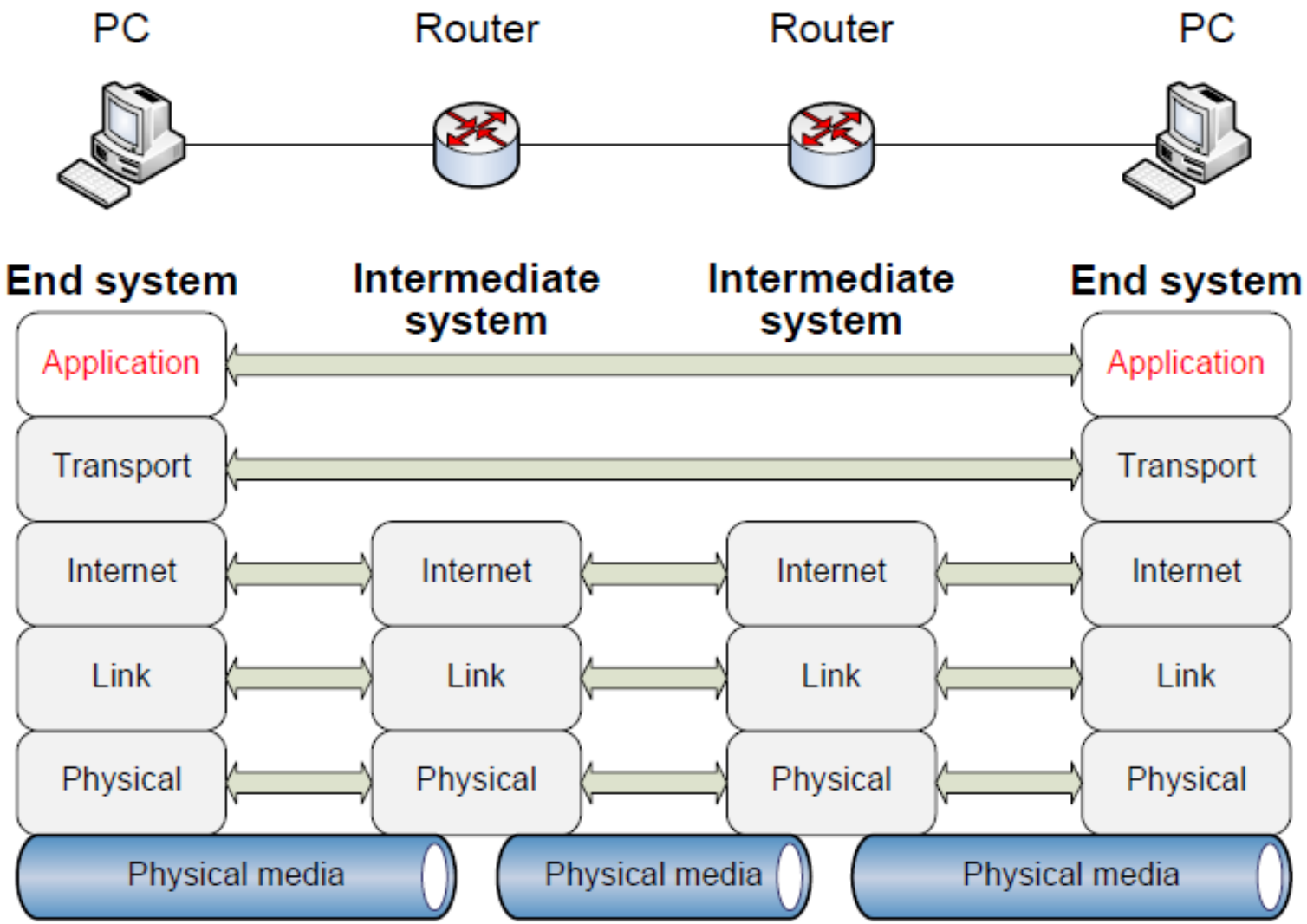


Fig. 1.17, Hwang, Fox, Dongarra Book

Peer-to-Peer Systems

P2P Systems' Protocols operate at the Application Layer of the TCP/IP Model

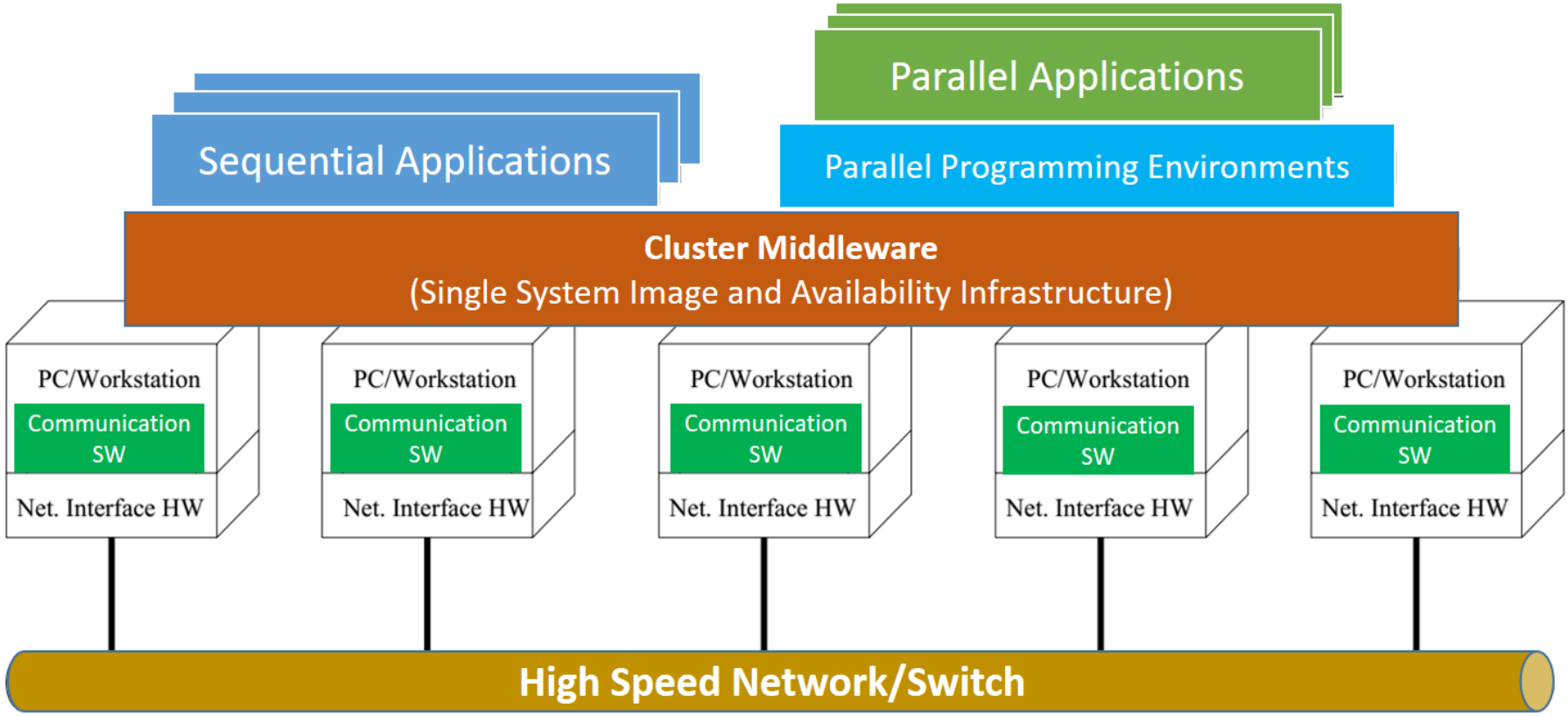


Peer-to-Peer Systems

System Features	Distributed File Sharing	Collaborative Platform	Distributed P2P Computing	P2P Platform
Attractive Applications	Content distribution of MP3 music, video, open software, etc.	Instant messaging, collaborative design and gaming	Scientific exploration and social networking	Open networks for public resources
Operational Problems	Loose security and serious online copyright violations	Lack of trust, disturbed by spam, privacy, and peer collusion	Security holes, selfish partners, and peer collusion	Lack of standards or protection protocols
Example Systems	Gnutella, Napster, eMule, BitTorrent, Aimster, KaZaA, etc.	ICQ, AIM, Groove, Magi, Multiplayer Games, Skype, etc.	SETI@home, Geonome@home, etc.	JXTA, .NET, FightingAid@home, etc.

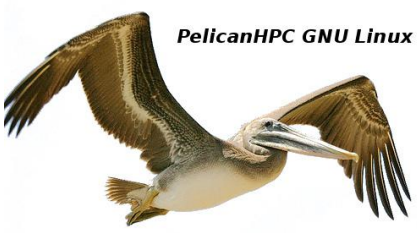
Table. 1.5, Hwang, Fox, Dongarra Book

Cluster Computing



Computer Cluster Architecture

Cluster Computing

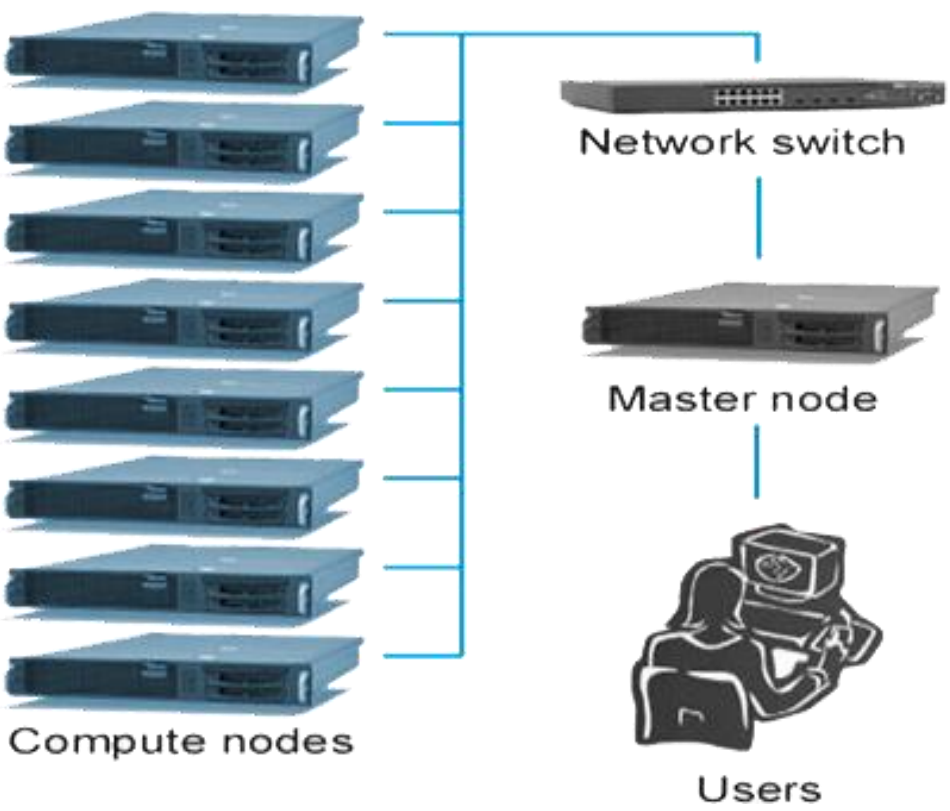


A Pelican Linux Cluster

Refer: https://pelicanhpc.org/

Cluster Computing

Cluster Computing



Cluster Computing

A Large Cluster Computer (Virtual Super Computer)

Cluster Computing – Types of Clusters

High Availability Clusters (HA) (Linux)

- Mission critical applications
- High-availability clusters (also known as Failover Clusters) are implemented for the purpose of improving the availability of services which the cluster provides.
- Provide Redundancy
- Eliminate Single Points of Failure.

Load Balancing Clusters

- Operate by distributing a workload evenly over multiple back end nodes.
- Typically the cluster will be configured with multiple redundant load-balancing front ends.
- All available Servers process requests.

Compute Clusters / High Performance Clusters

- Designed for collective computation of a single task.
- **Beowulf**

Grid Computing

- The term *Grid* comes from an analogy to the Electricity Grid.
  - Pervasive access to power.
- Grid Computing refers to pervasive, consistent, and inexpensive access to advanced computational resources.
- Grid computing is all about achieving greater performance and throughput by pooling and sharing resources & information on a local, national, or international level.

Grid Computing

- Grids are about large-scale resource sharing.
  - *Spanning administrative boundaries.*
    - Central processors, storage, network bandwidth, databases, applications, instruments, sensors and so on… … …
- Problem solving in a dynamic, *multi-institutional environment*.
- Organizing *geographically distributed* computing resources
  - So that they can be flexibly and dynamically allocated and accessed
- Sharing is highly controlled, clear definitions of exactly what is shared, who is allowed to share, and the conditions under which sharing occurs.

Grid Computing – Main Applications

- **Resource Sharing**
  - Computers, data, storage, sensors, networks, …
  - Sharing always conditional: issues of trust, policy, negotiation, payment, …
- **Coordinated Problem Solving**
  - Beyond client-server: distributed data analysis, computation, collaboration, …
- **Dynamic, Multi-institutional *Virtual Organizations***
  - Community overlays on classic org. structures
  - Large or small, static or dynamic

Grid Computing – *Virtual Organizations (VO's)*

- A set of individuals and/or institutions defined by a set of sharing rules and policies.
- The sharing is highly controlled, with resource providers and consumers defining clearly and carefully what is shared among whom.
- CPU Scavenging - create Virtual Supercomputers using unused resources in a network of existing computers.
- VO offers dynamic cooperation built over multiple physical organizations.

Grid Computing – *Virtual Organizations (VO's)*

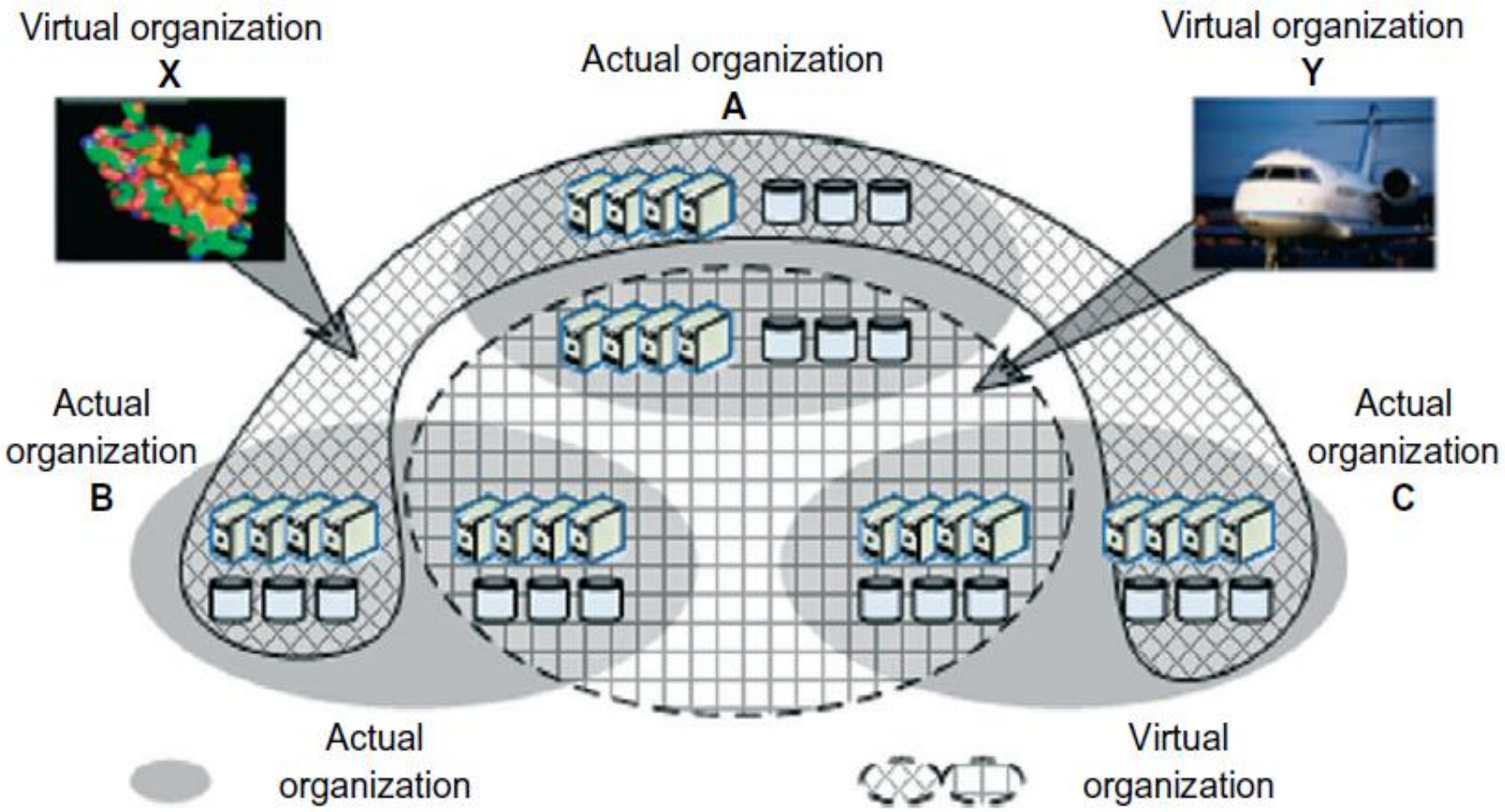


Fig. 7.2 (pg. 421) : Hwang, Fox, Dongarra Book

Grid Computing - Benefits

- Exploiting under-utilized resources
- Obtaining massively parallel CPU capacity
- Virtual Resources & Virtual Organizations
- Access to expensive resources
- Resource Balancing
- Reliability

Utility Computing

- A Service Provisioning Model
  - Service Provider provides "on demand"
    - Computing Resources – CPU, Storage, APIs, Bandwidth… …
    - Infrastructure Management… … and so on.
  - Charges for specific usage than flat rate
- The term **utility** refers to packaging of system resources as a **metered** service.
- Built with multiple back-end servers, dedicated clusters, or even under-utilized super computers
- Advantage: Low or Zero Initial Cost, pay for use
- Paradigm shift from **"Product"** to **"Service"**

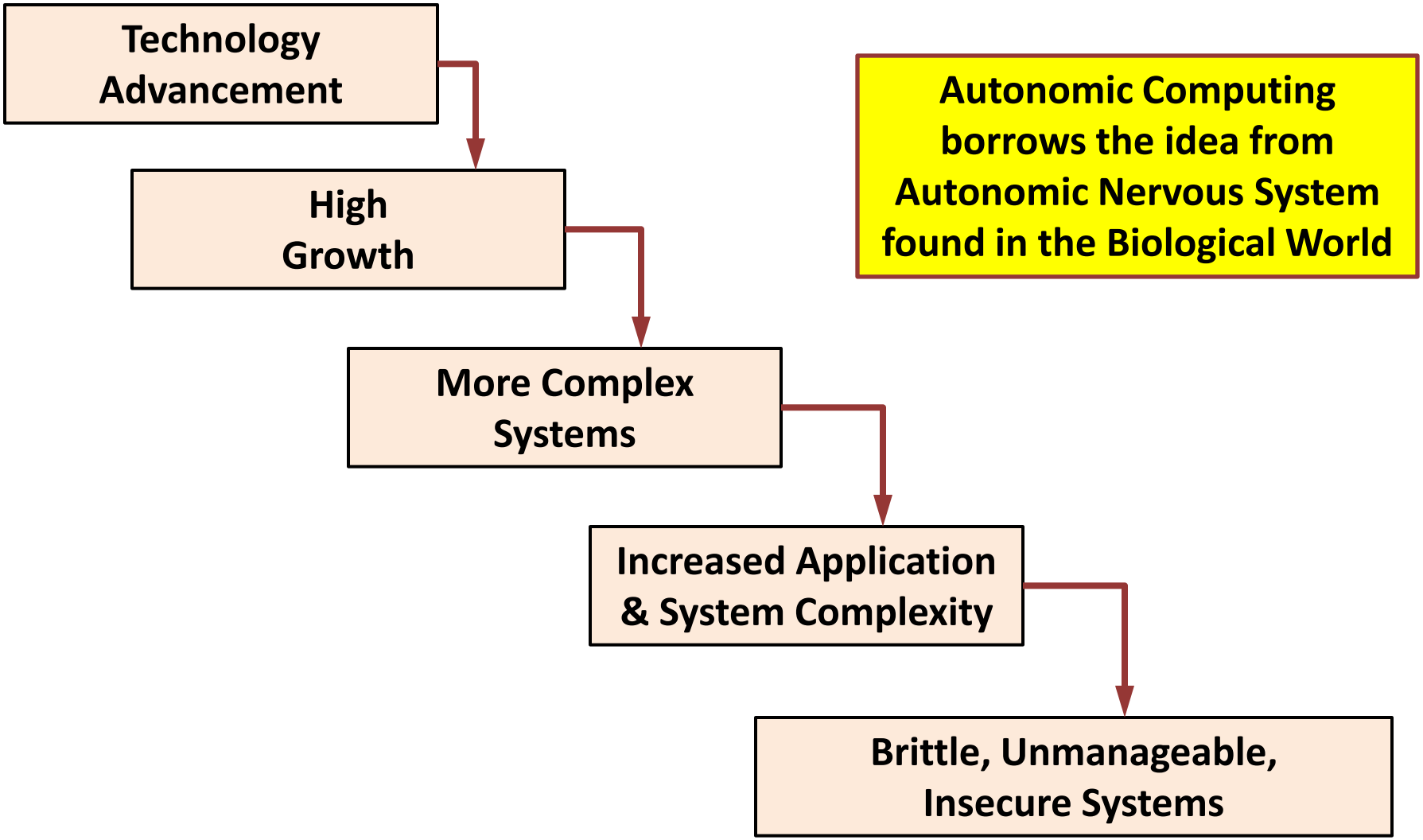
Autonomic Computing

"A self-managing autonomous and ubiquitous computing environment that completely hides its complexity, thus providing the user with an interface that exactly meets the needs at all points of time"

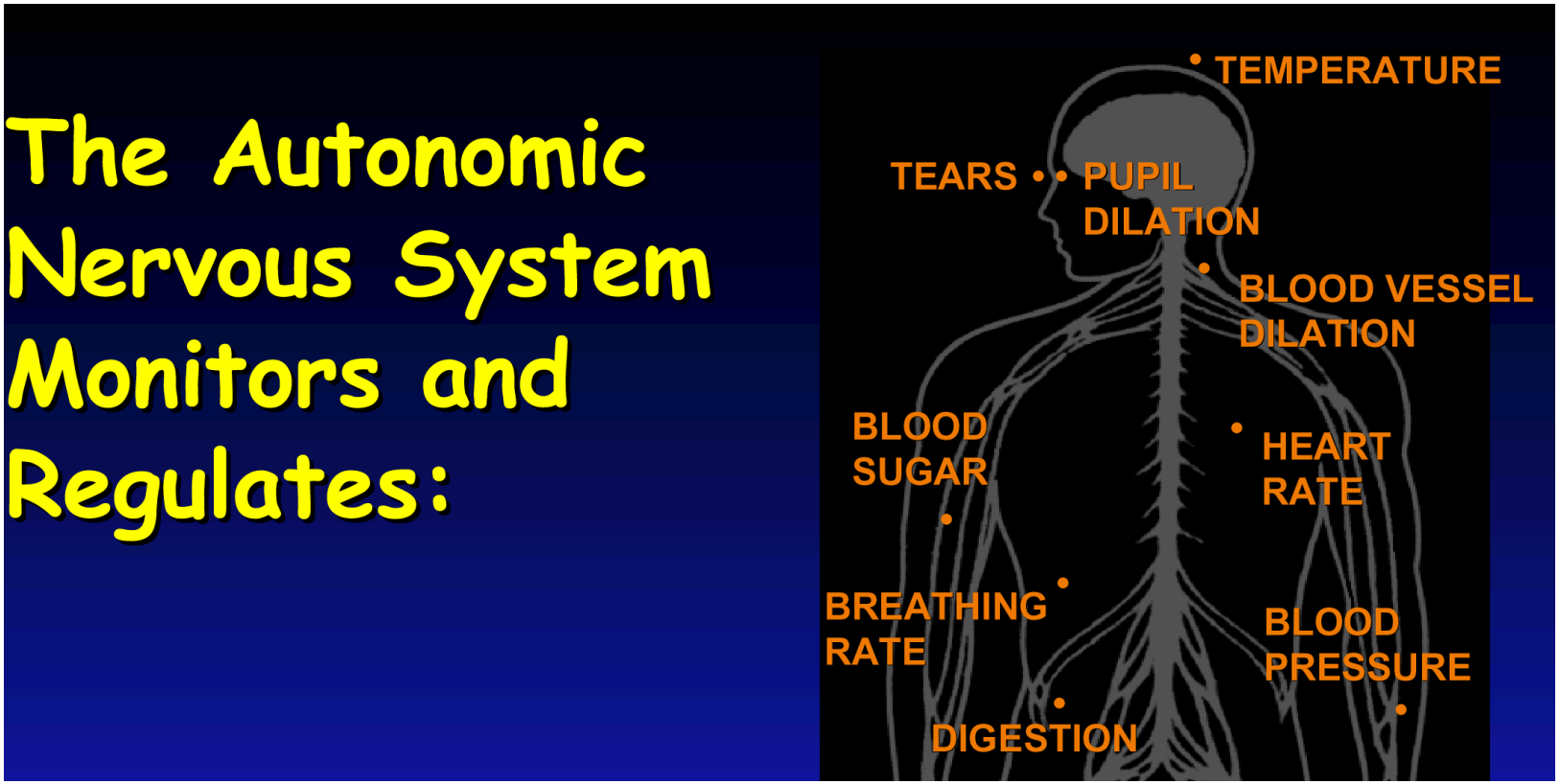
Proposed by IBM in 2001

Refers to <u>self-managing</u> characteristics of distributed computing resources and adapting to unpredictable changes to <u>overcome the complexity</u> of ever-growing IT Infrastructure to operators and users.

Why Autonomic Computing?



Autonomic Computing



Without requiring our conscious involvement

Examples:

When we run, it increases our heart and breathing rate.

When there is less light, our pupils dilate.

Autonomic Computing

Self-protecting System designed to protect itself from any unauthorized access anywhere

Self-optimizing System designed to automatically manage resources to allow the servers to meet the enterprise needs in the most efficient fashion

Self-healing
Autonomic problem determination and resolution

Self-configuring
systems designed to define itself "on the fly"

Autonomic Computing - Characteristics

Increased Responsiveness

Adapt to dynamically changing environments

Business Resiliency

Discover, diagnose, and act to prevent disruptions

Operational Efficiency

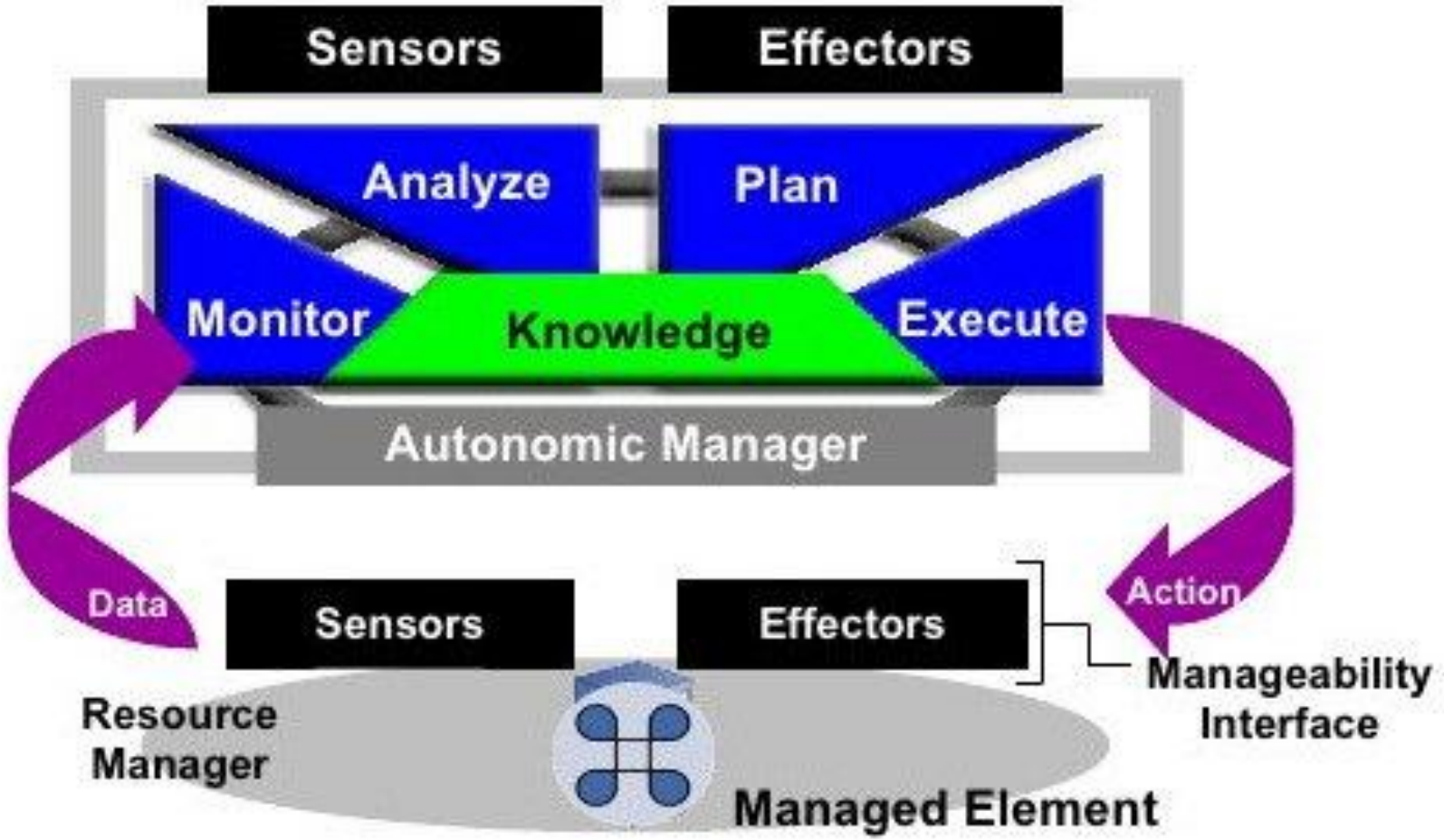
Tune resources and balance workloads to maximize use of IT resources

Secure Information and Resources

Anticipate, detect, identify, and protect against attacks

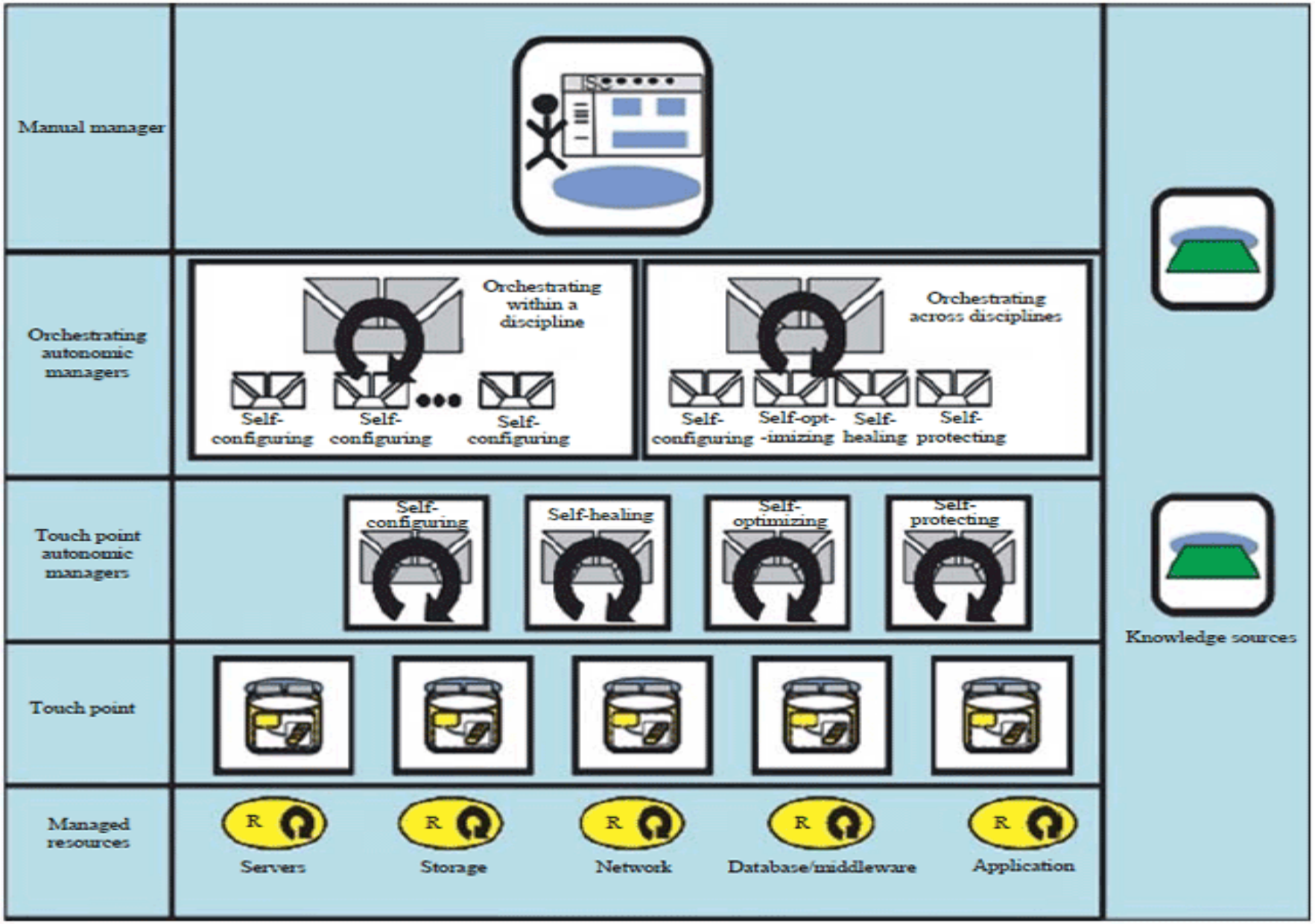
These are also the "benefits" of Autonomic Computing Systems

Autonomic Computing

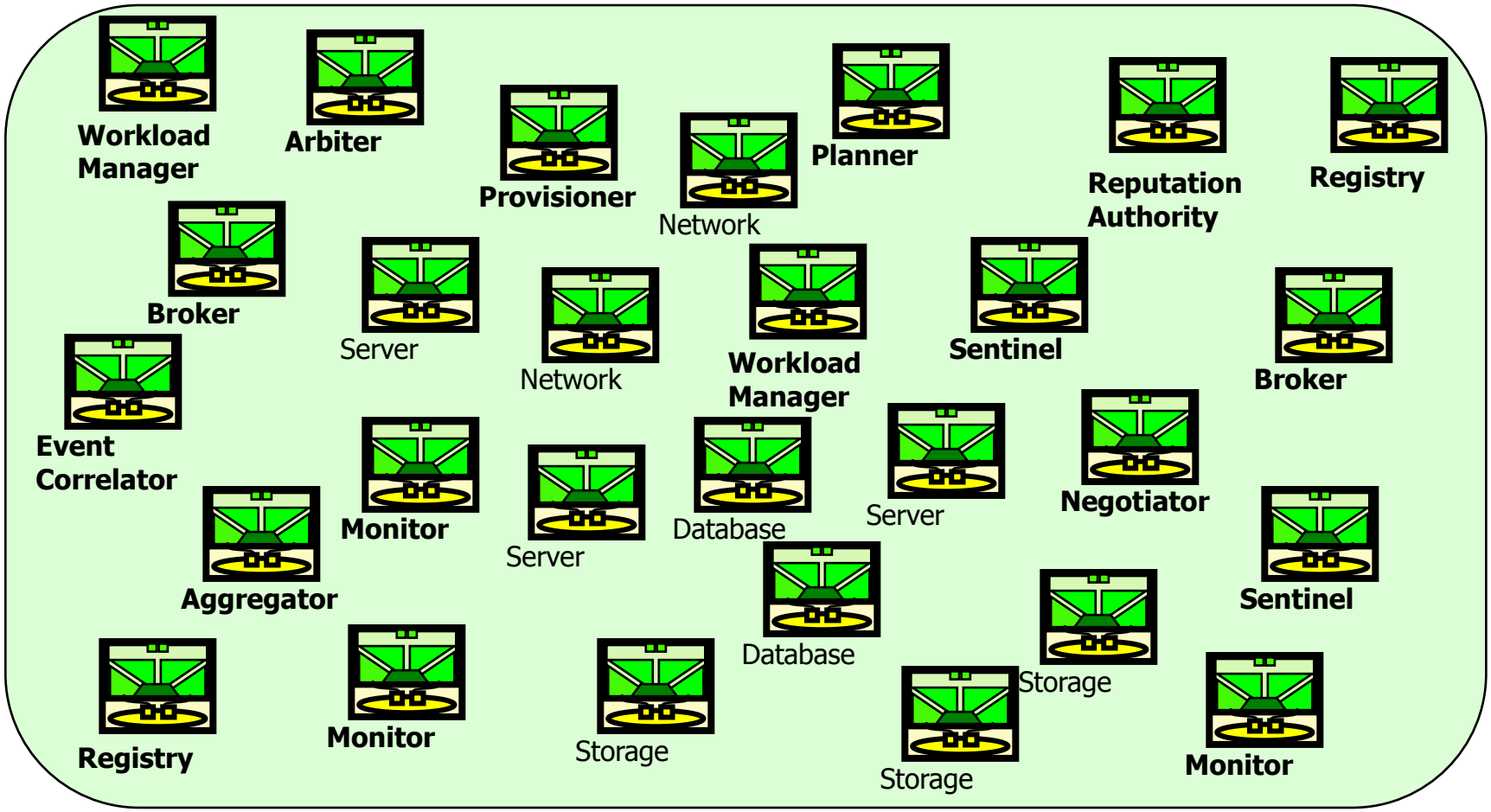


Managed Element: ***Fundamental ATOM of Autonomic Computing Architecture***

Autonomic Computing Architecture (IBM)



Autonomic Computing



Composition of Autonomic Elements

Autonomic Computing

- Berkeley OceanStore project
  - http://oceanstore.cs.berkeley.edu
- IBM's StorageTank
  - Multiplatform universally accessible storage mgmt.
- Oceano
  - http://www.research.ibm.com/oceanoproject/
  - Management of Software farms
  - Now conducted at IBM's Hafia Lab
- IBM Smart DB2
  - Reduction of human intervention of database server management costs

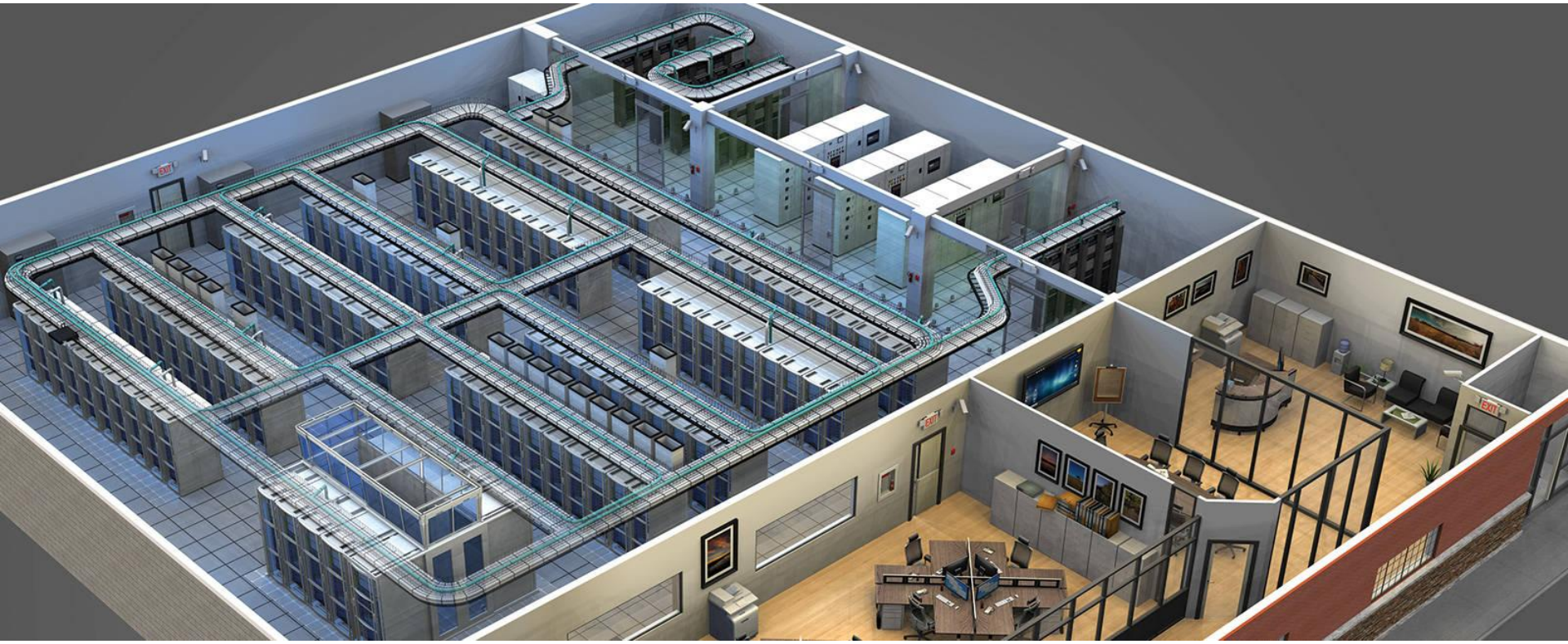
Hosting (Web Hosting)

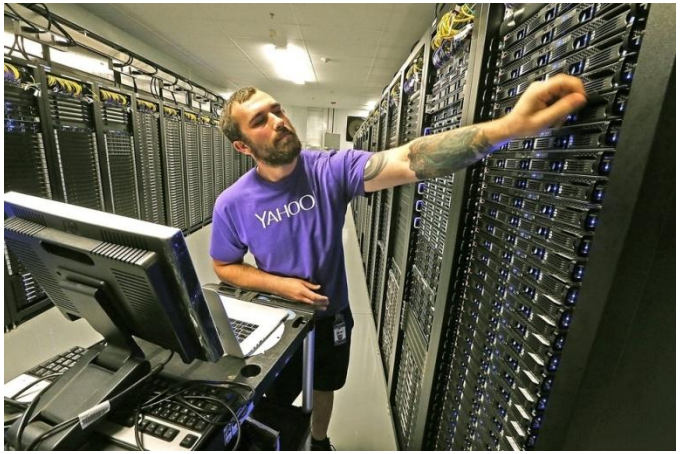
Types of Hosting

- Shared Hosting
- Reseller Hosting
- Virtual Dedicated Server
- Dedicated Server (*with root access)*
- Managed Hosting (*without root access)*
- Colocation Hosting
- Clustered Hosting
- Grid Hosting
- Cloud Hosting

Data Center

Data Center

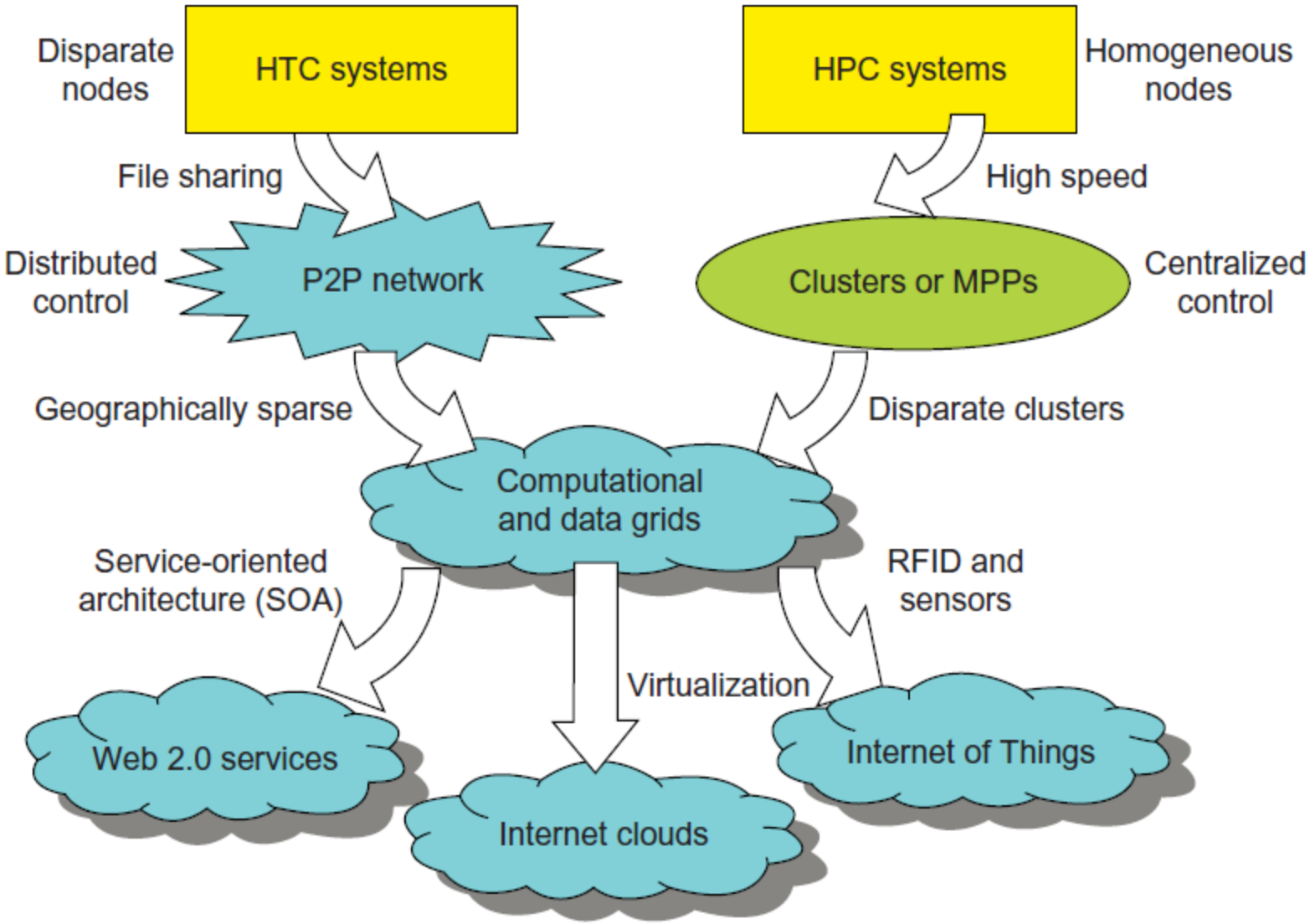




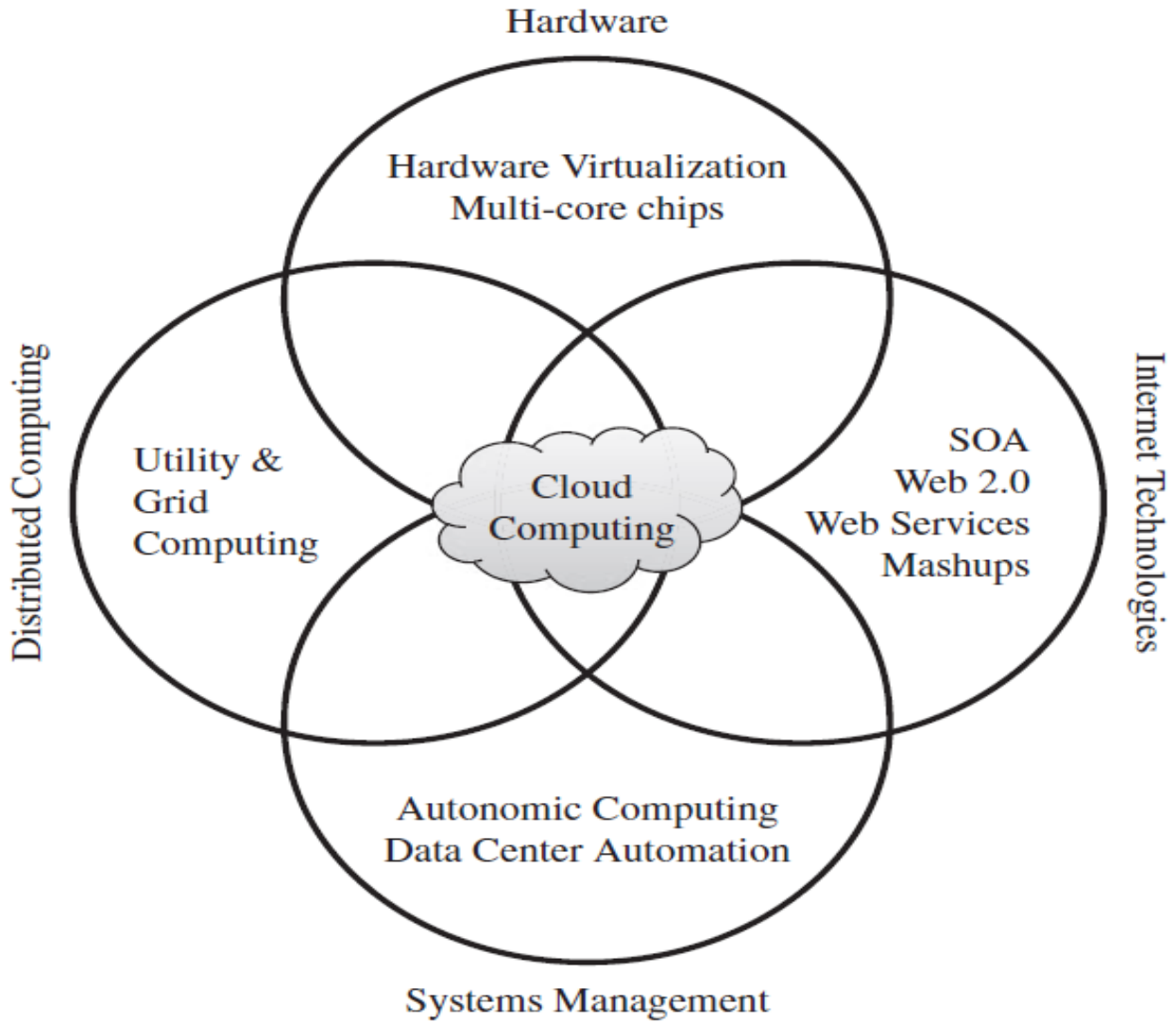
Data Center

Google data center in the Dalles, Oregon, USA ($1.8 billion)
(the first data center owned & operated by Google)

The Platform Evolution

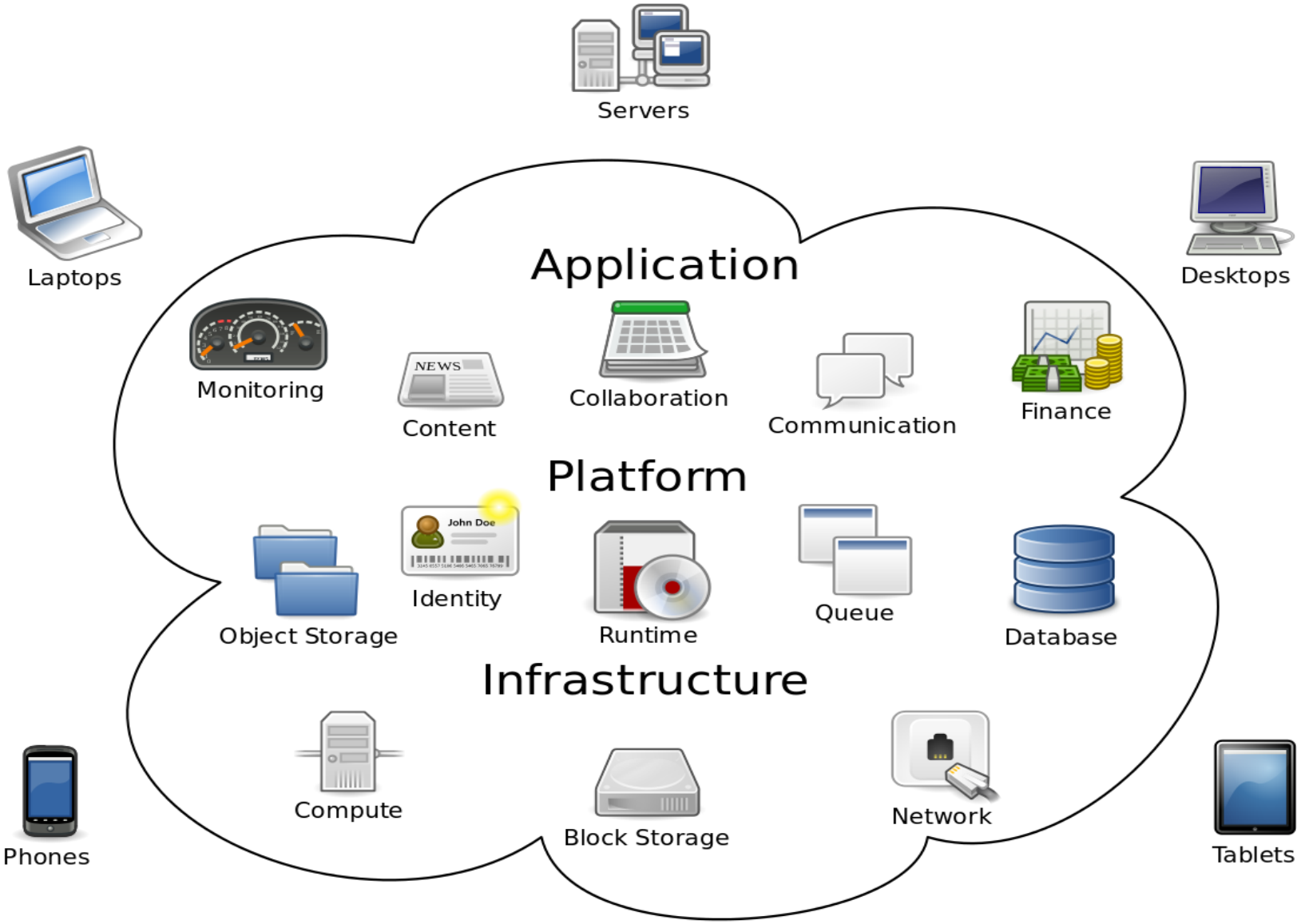


Cloud Computing



Convergence of Various Technological Advancements has led to Cloud Computing

Cloud Computing



Cloud Computing

- **C** – Common
- **L** – Location Independent
- **O** – Online *(i.e., over the Internet)*
- **U** – Utility *(that is available on…)*
- **D** – Demand

Note: This is a manufactured full form. The term "CLOUD" is not an abbreviation, and used as a metaphor to represent computing over the Internet.

Cloud Computing

NIST (National Institute of Standards & Technology, USA) defines Cloud Computing as:

"Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction."

The NIST Model

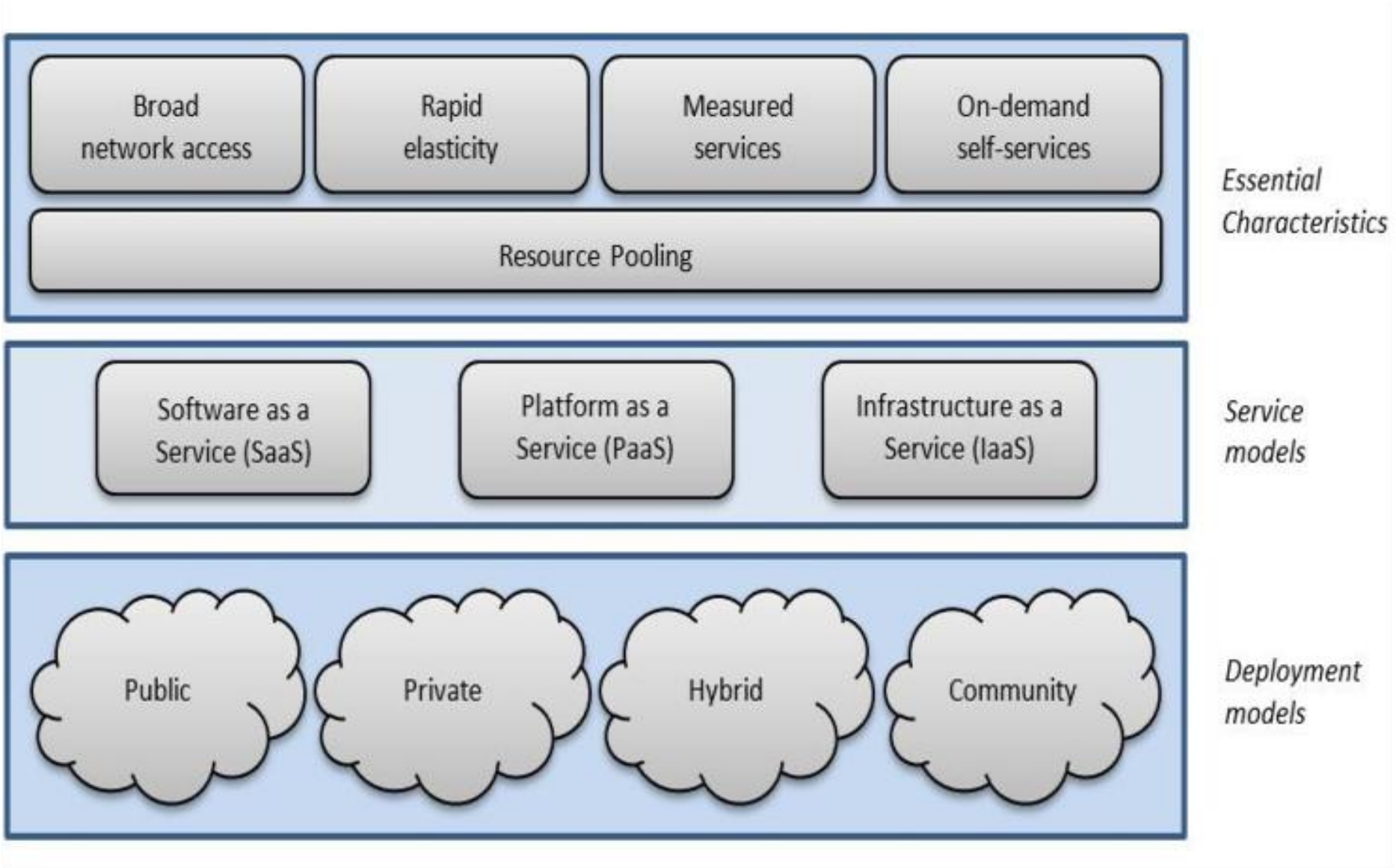
- The term “cloud” refers to 2 important concepts
  - **Abstraction**
  - **Virtualization**

- The NIST Model of Cloud Computing consists of:
  - **5** Essential Characteristics
  - **3** Service Models
  - **4** Deployment Models

Common Characteristics

- The NIST Model also defines 8 common/general characteristics of Cloud Computing
  - Massive Scale
  - Resilient Computing
  - Homogeneity
  - Geographic Distribution
  - Virtualization
  - Service Orientation
  - Low Cost Software
  - Advanced Security

Cloud Computing – The NIST Model



Cloud Deployment Models

Operated solely for a single organization

Maybe on premise or off premise

Shared by several entities that have a common purpose.

Maybe on premise or off premise

Available to the general public and owned by a single organization selling cloud services.

Any combination of two or more private / community or public clouds.

Cloud Deployment Models

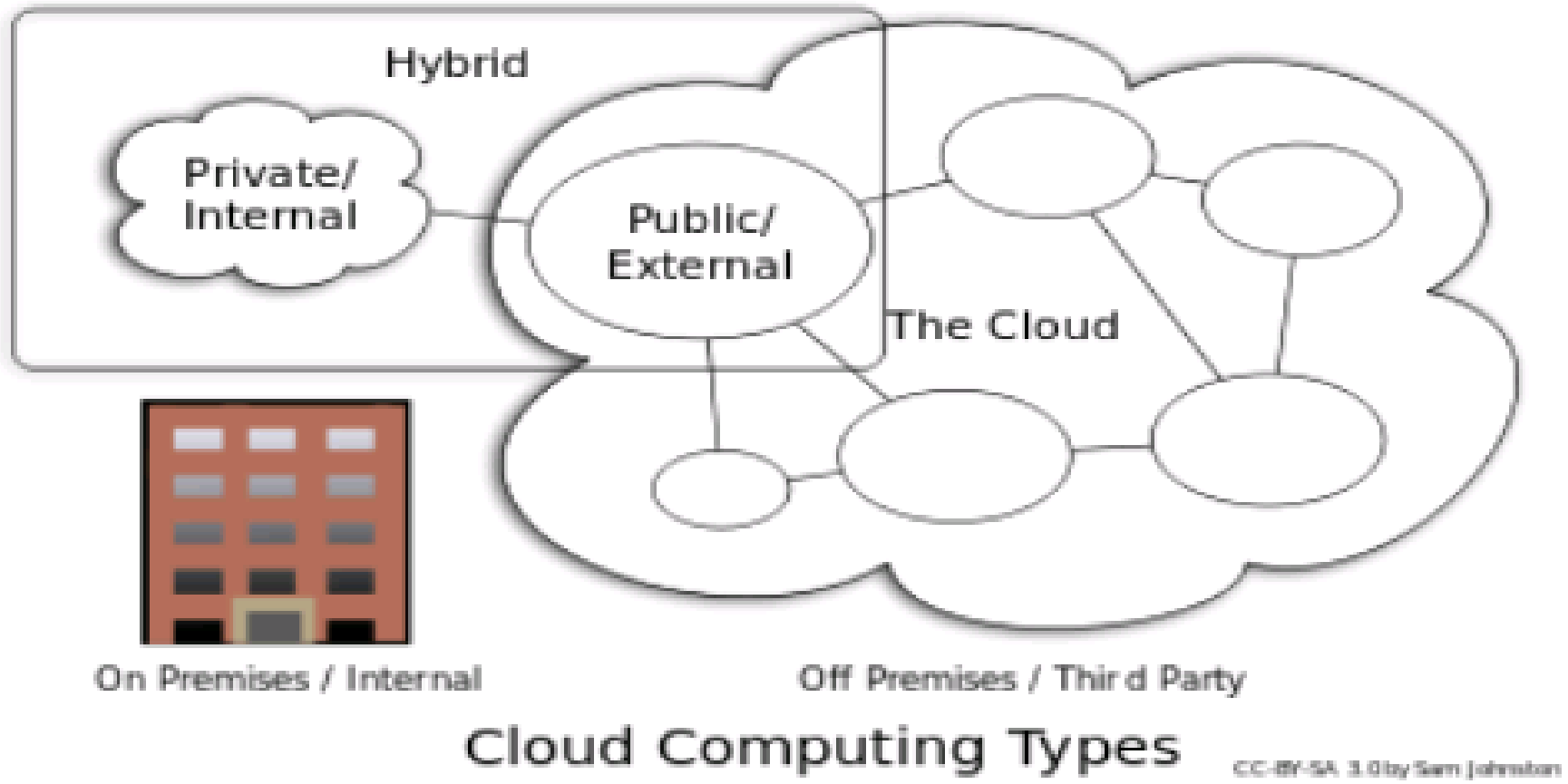
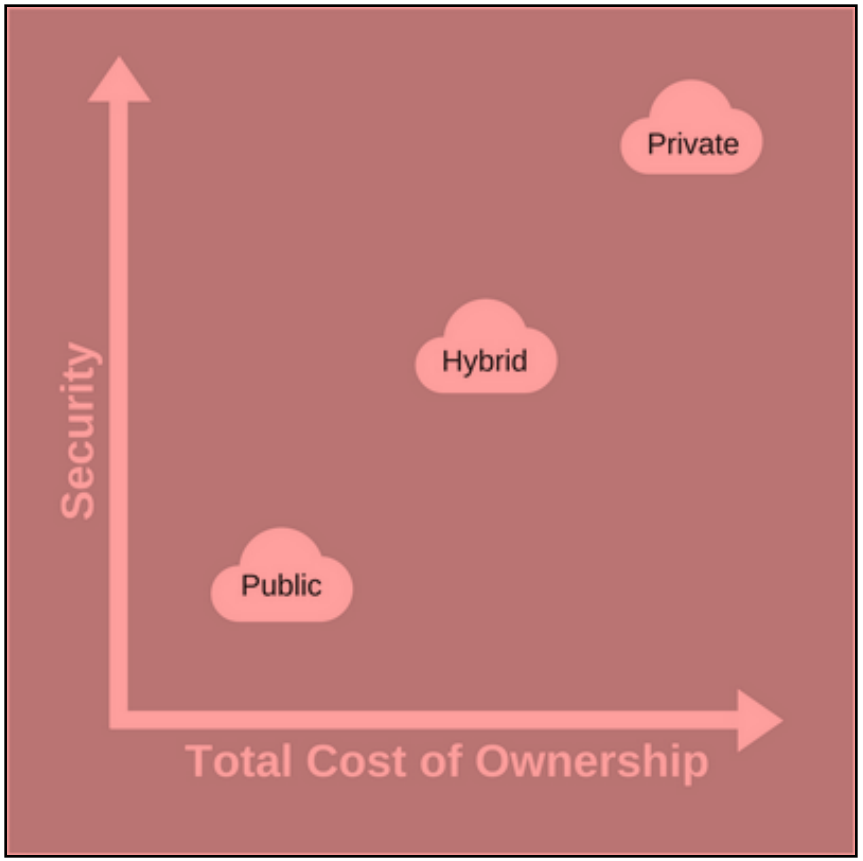
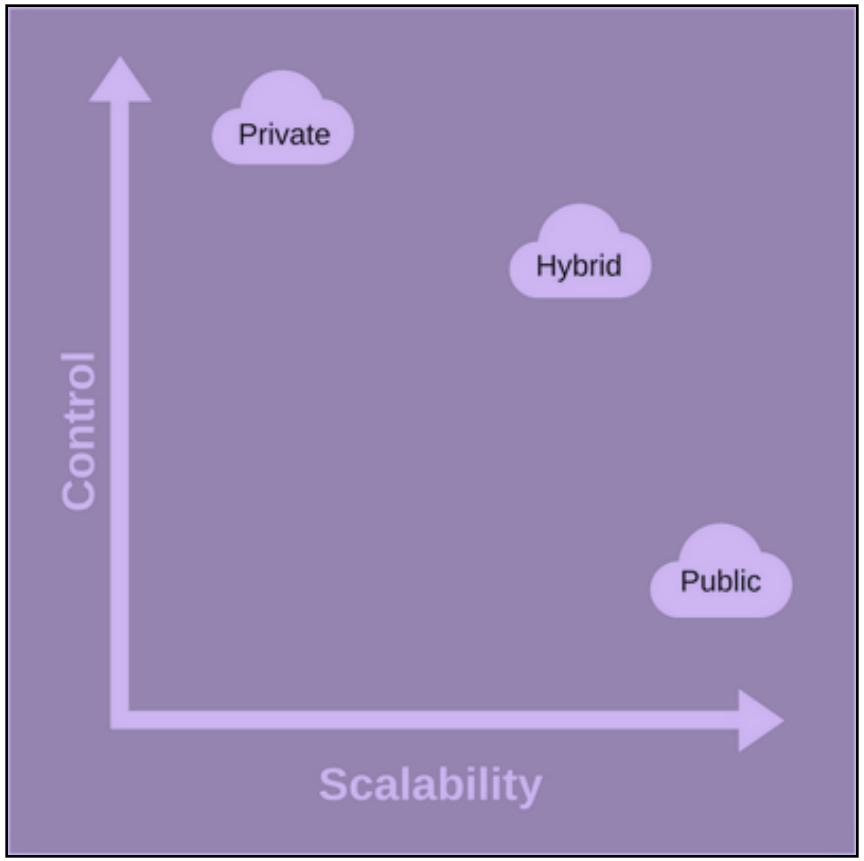


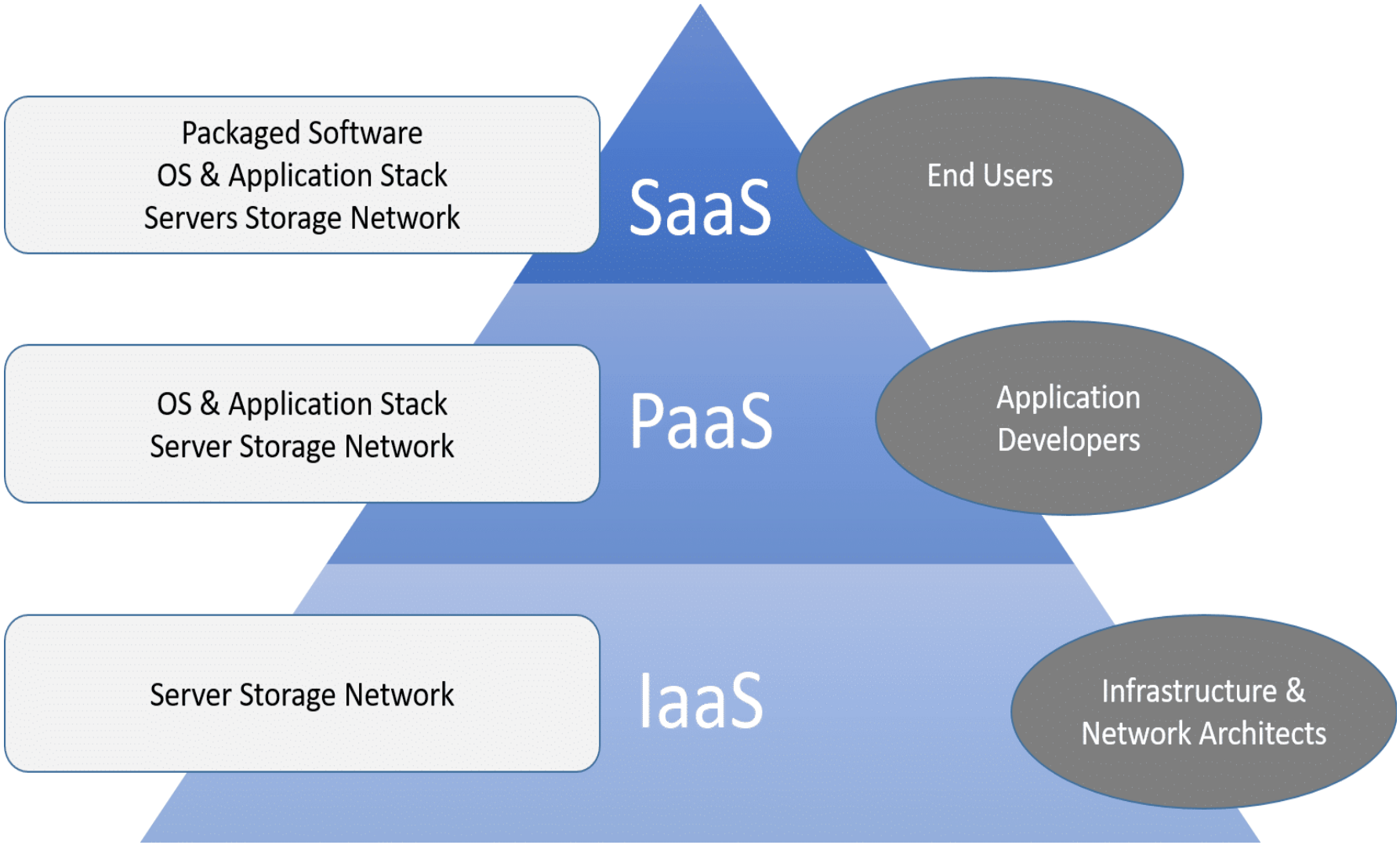
Fig: deployment models of cloud computing

Comparison of Deployment Models





Cloud Service Models



Infrastructure as a Service (IaaS)

- Consumer provisions processing, storage, networks, and other fundamental computing resources
- Consumer does not manage or control the underlying cloud infrastructure
- Consumer is able to deploy and run arbitrary software (in/c OS & Applications)
- Consumer may have limited control of select networking components (e.g., host firewalls).

Platform as a Service (PaaS)

- Consumer is able to deploy own or acquired applications developed using programming languages, libraries, services, and tools which are supported by the provider.
- Consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage,
- Consumer has full control over the deployed applications and possibly configuration settings for the application-hosting environment.

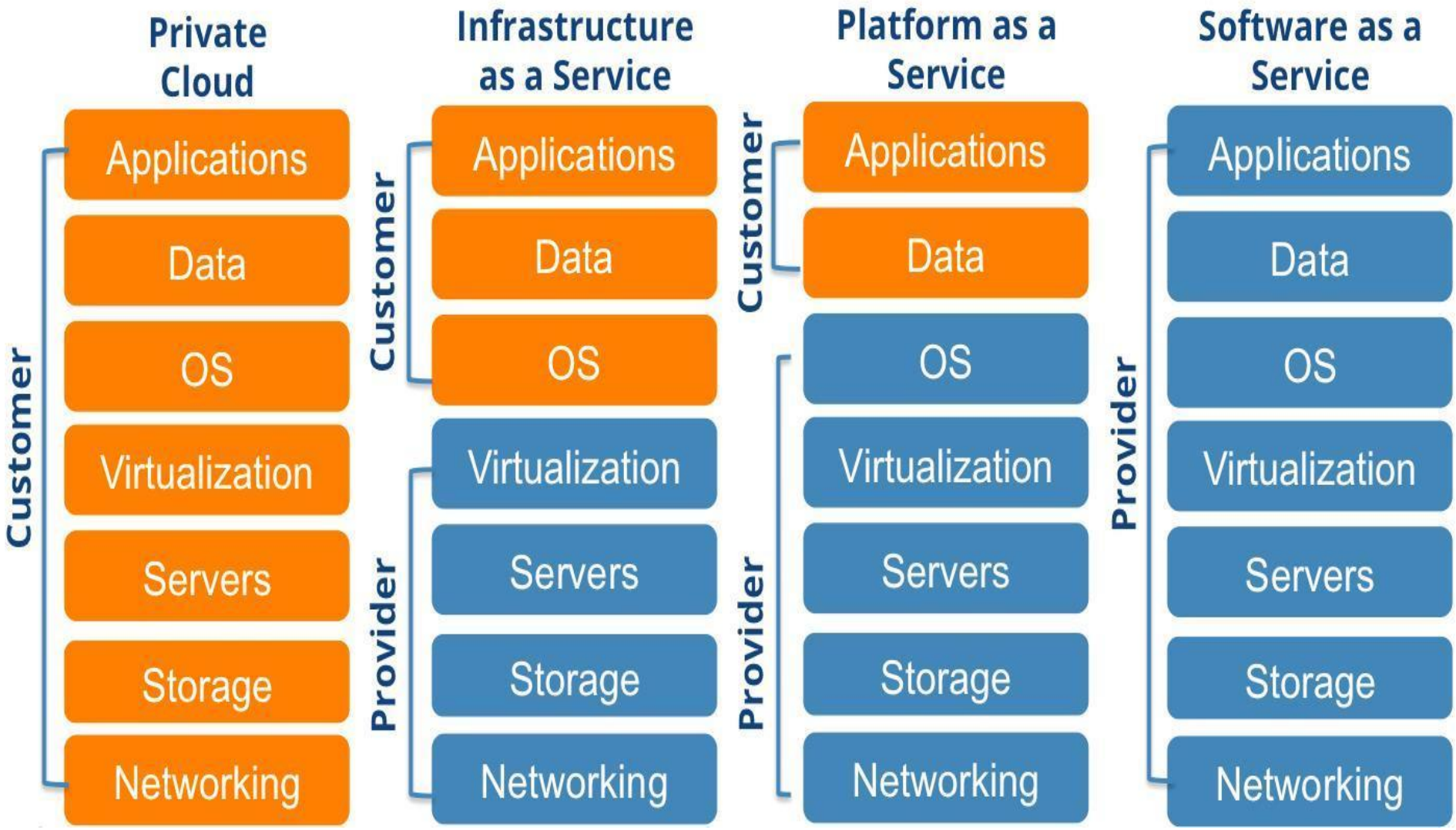
Software as a Service (SaaS)

- Consumer is able to use the provider's applications running on a cloud infrastructure.
- The applications are accessible from various client devices using a thin client or a program interface.
- The consumer does not manage or control the underlying cloud infrastructure or individual application capabilities
- Consumer may have limited control over some user-specific application configuration settings.

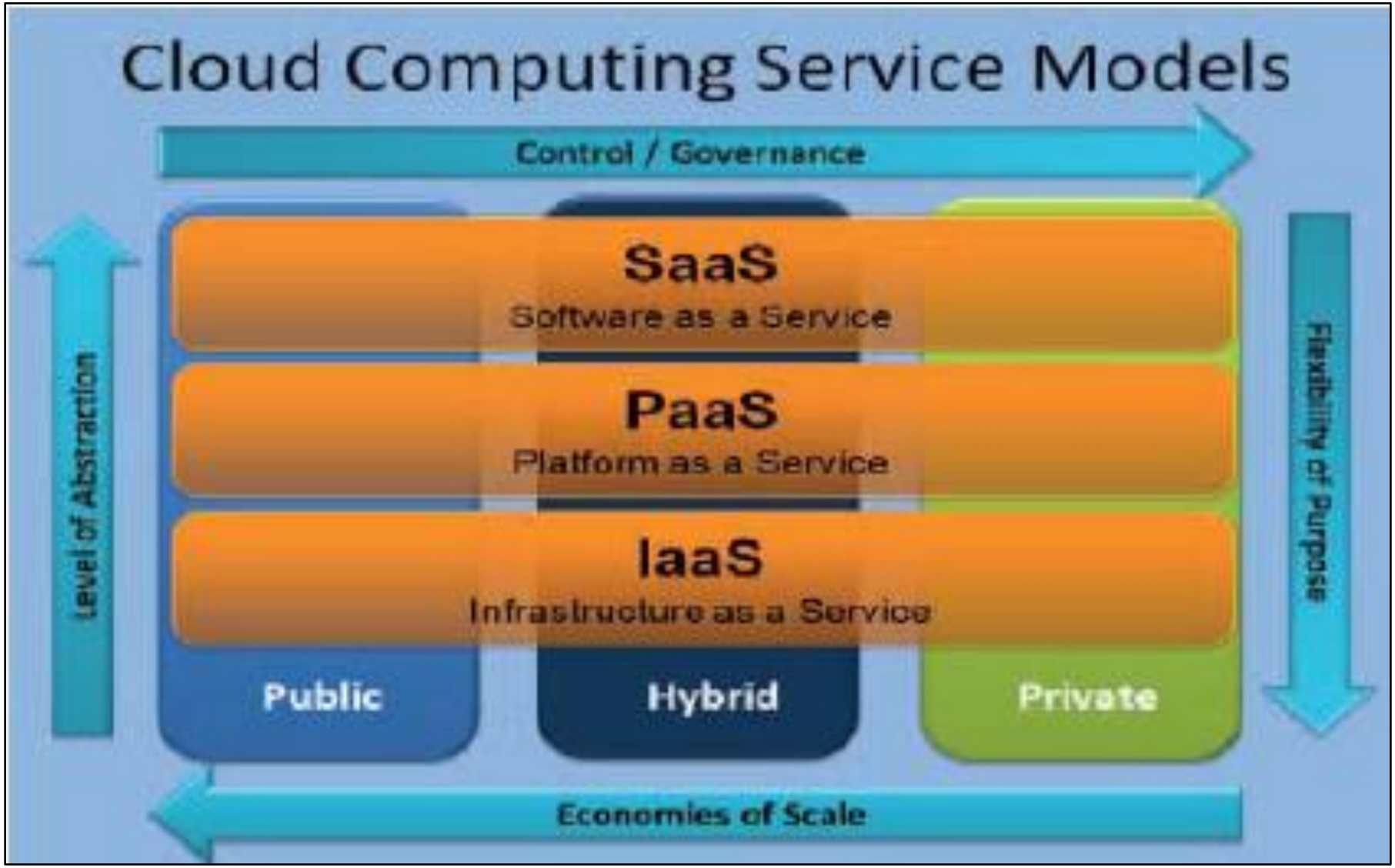
Comparison of Service Models

Pizza as a Service

Comparison of Cloud Service Models



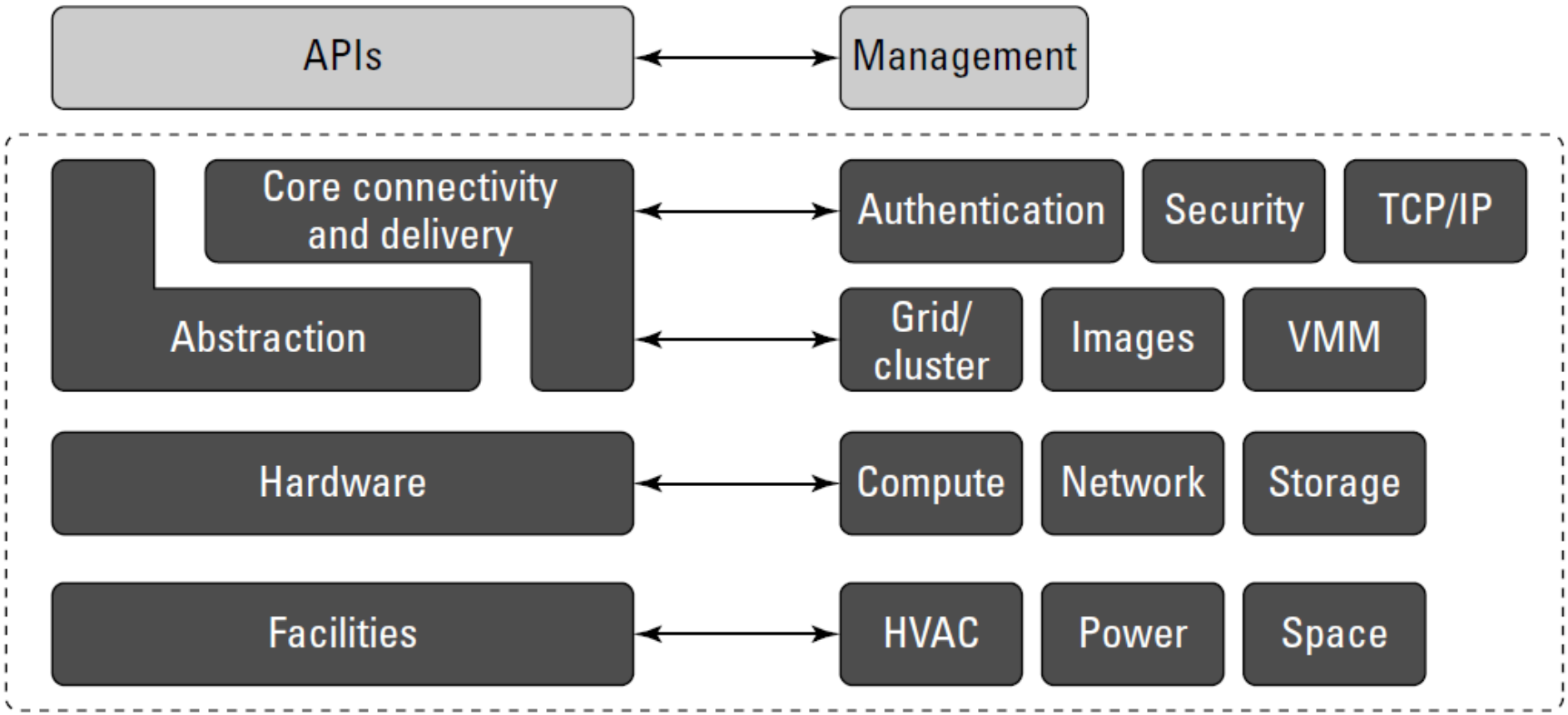
Economics of Cloud



Examples

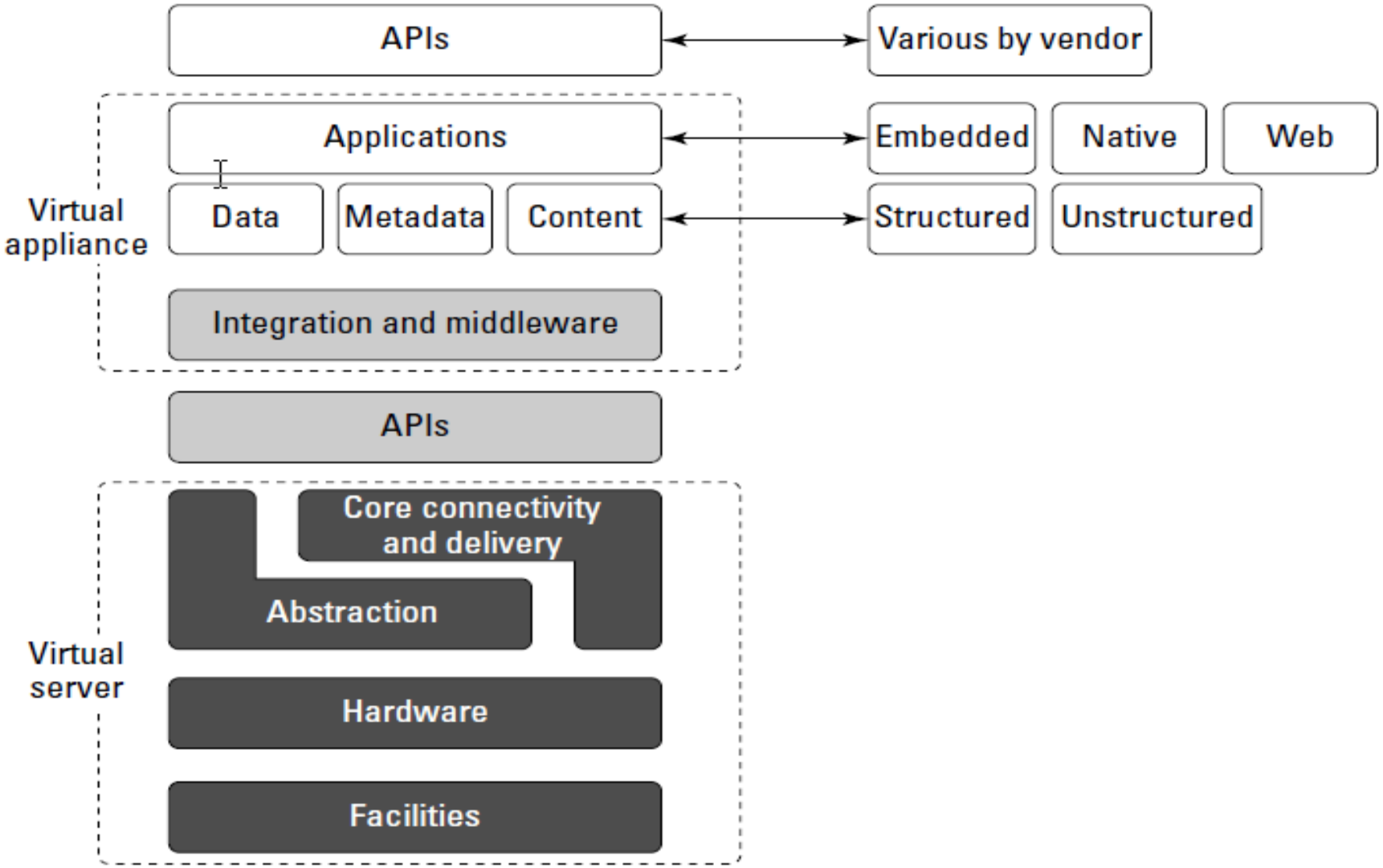
Type	Common Examples
IaaS	DigitalOcean, Linode, Rackspace, Amazon Web Services (AWS), Cisco Metapod, Microsoft Azure, Google Compute Engine (GCE)
PaaS	AWS Elastic Beanstalk, Windows Azure, Heroku, Force.com, Google App Engine, Apache Stratos, OpenShift
SaaS	Google Apps, Dropbox, Salesforce, Cisco WebEx, Concur, GoToMeeting

The Cloud Computing Stack



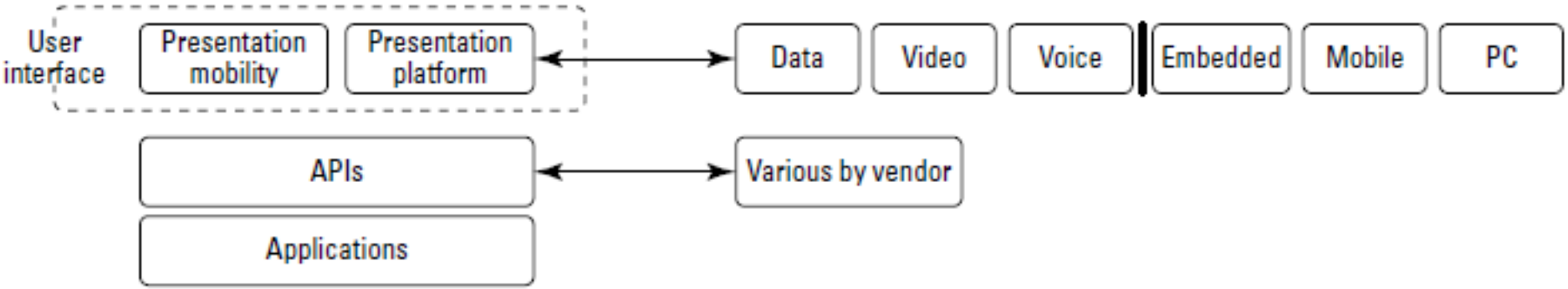
This architectural diagram illustrates the bottommost layer of the cloud computing stack that is designated as the **Virtual Server**

The Cloud Computing Stack



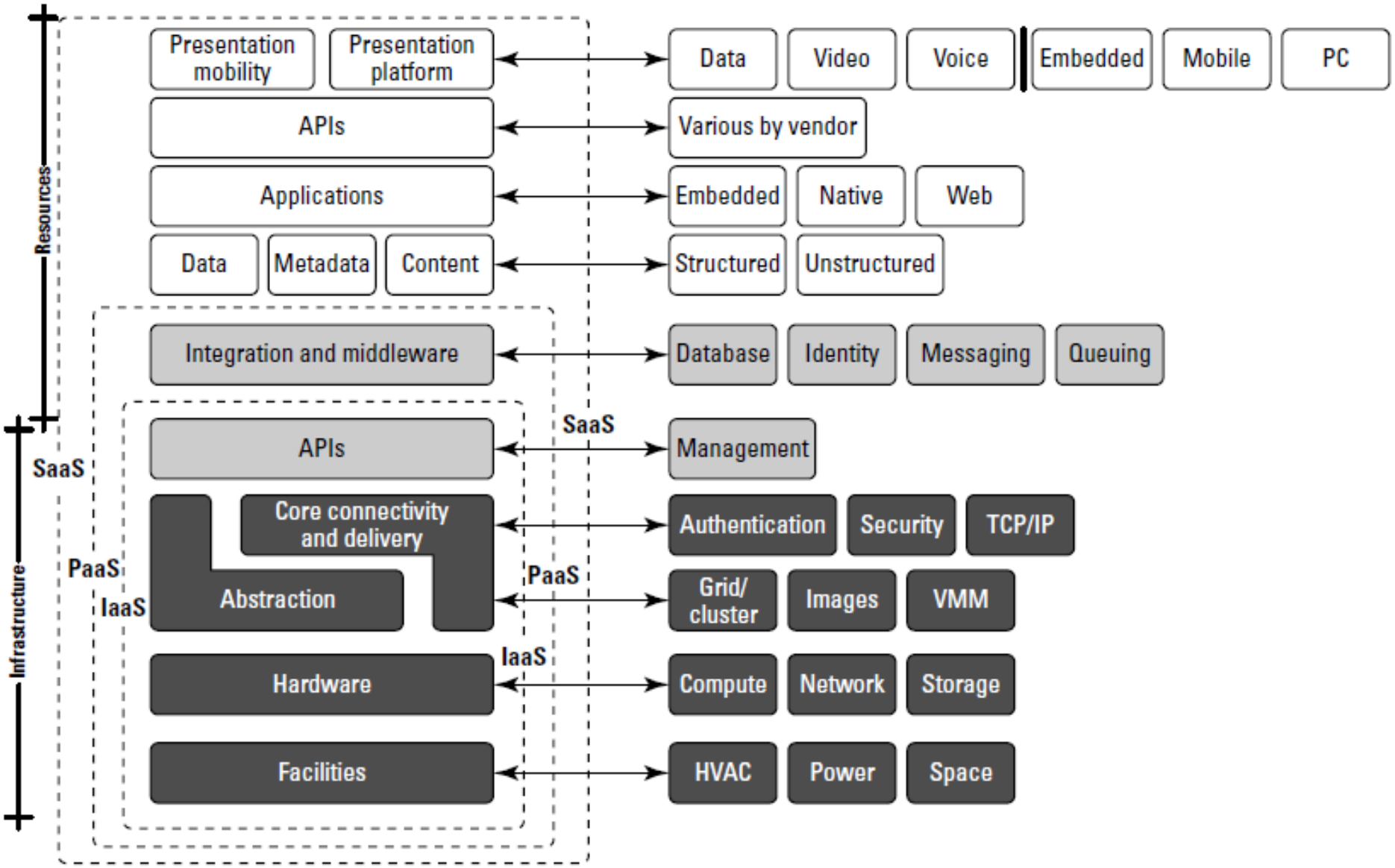
The middle layer is called a **Virtual Appliance** which is a software that installs as middleware onto a **Virtual Server**

The Cloud Computing Stack



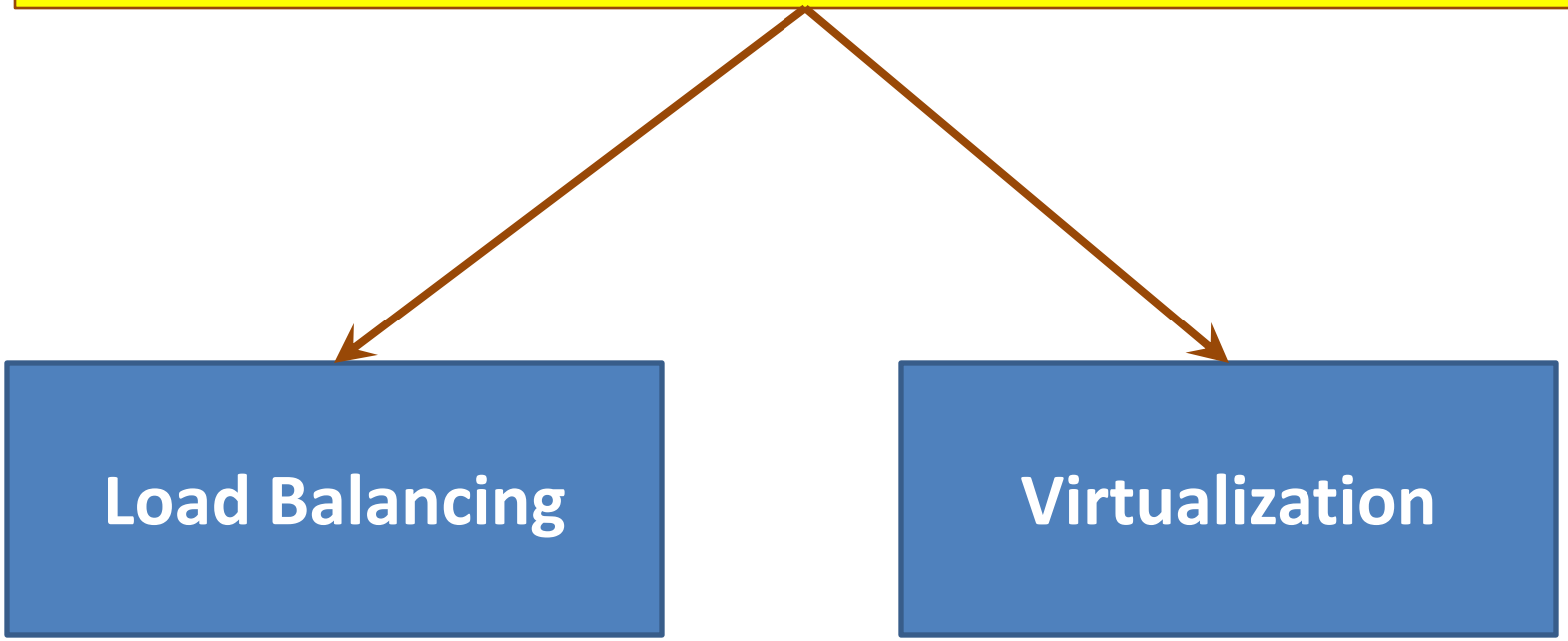
The topmost layer of the cloud computing stack includes the user interface and the APIs for the Application Layer

The Cloud Reference Model



Cloud Service Models (IaaS, PaaS, SaaS) w.r.t the Could Computing Stack

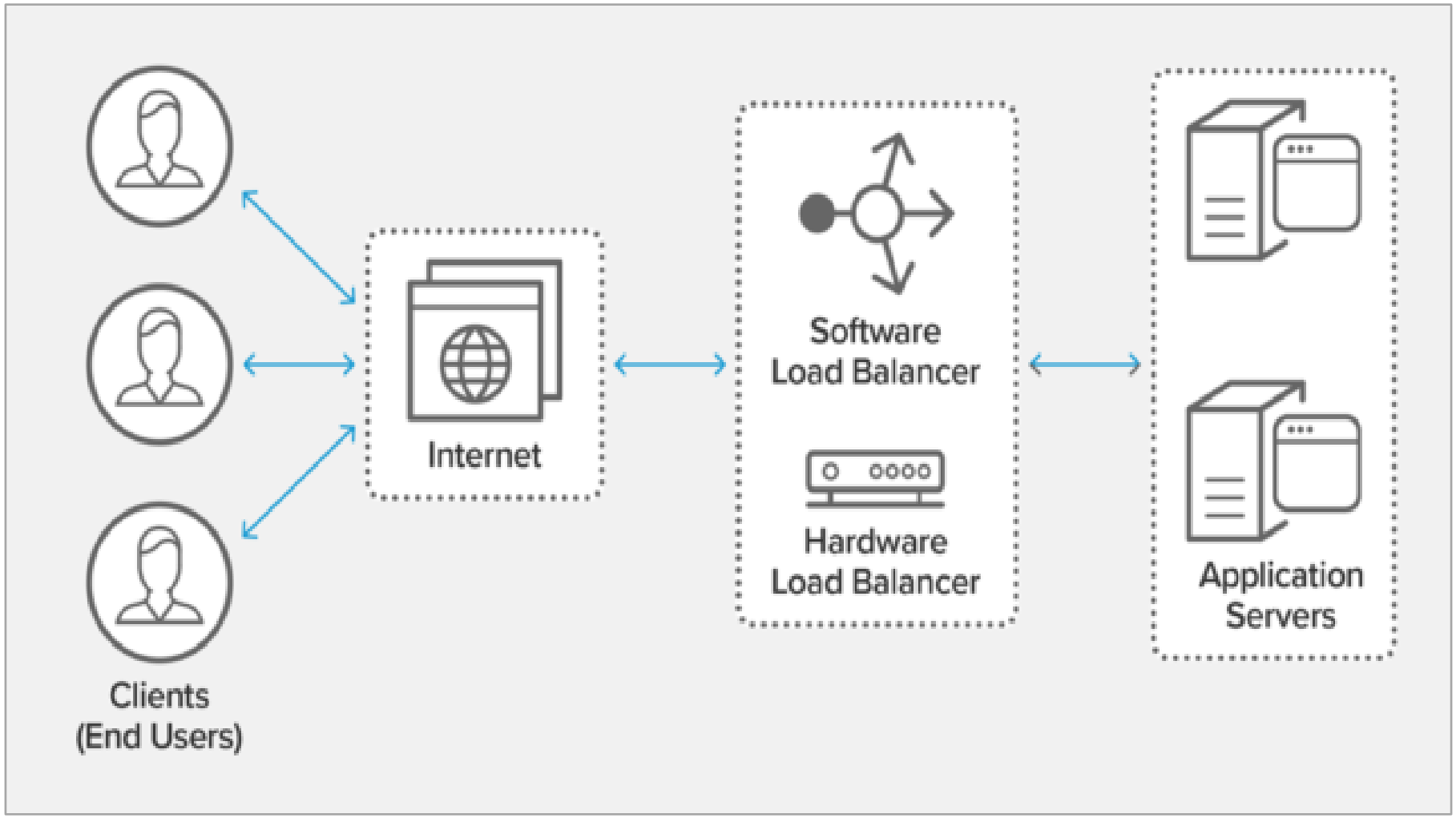
Cloud Computing is centered around two important concepts



Load Balancing

- Technology to distribute user requests (network traffic) to individual servers → Load Balancing
- Load balancing ensures even distribution of workload between the servers
- Useful for optimal use of resources in back-end
- Two Types
  - **Hardware Load-balancing Devices (HLD)**
    - Eg. F5 BigIP Server
  - **Software-based Load Balancing (SLB)**
    - Eg. Apache mod_proxy_balancer, Ngnix, Varnish, HAProxy

Load Balancing

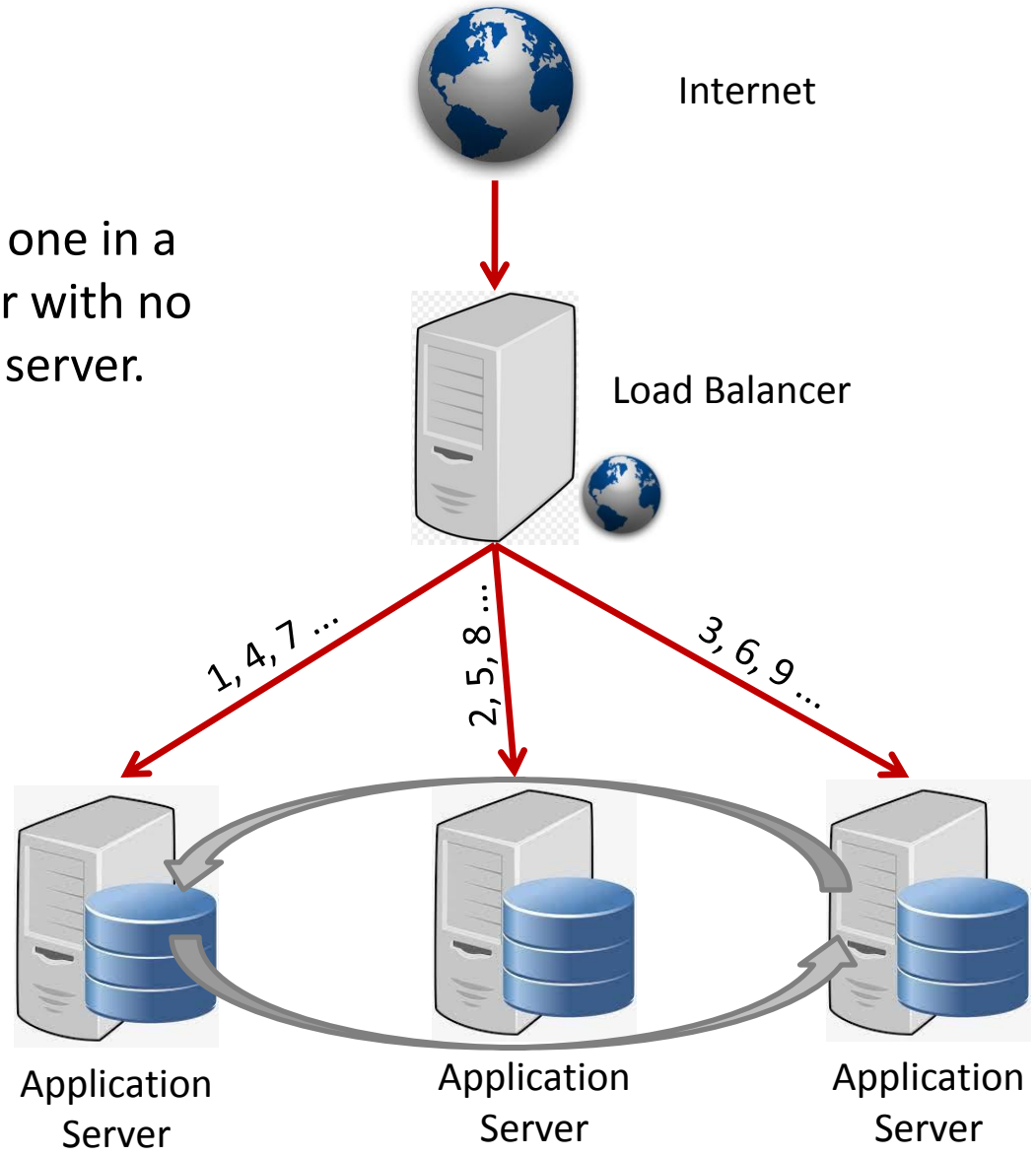


Load Balancing Algorithms

- Round Robin
- Weighted Round Robin
- Low Latency
- Least Connections
- Priority
- Overflow

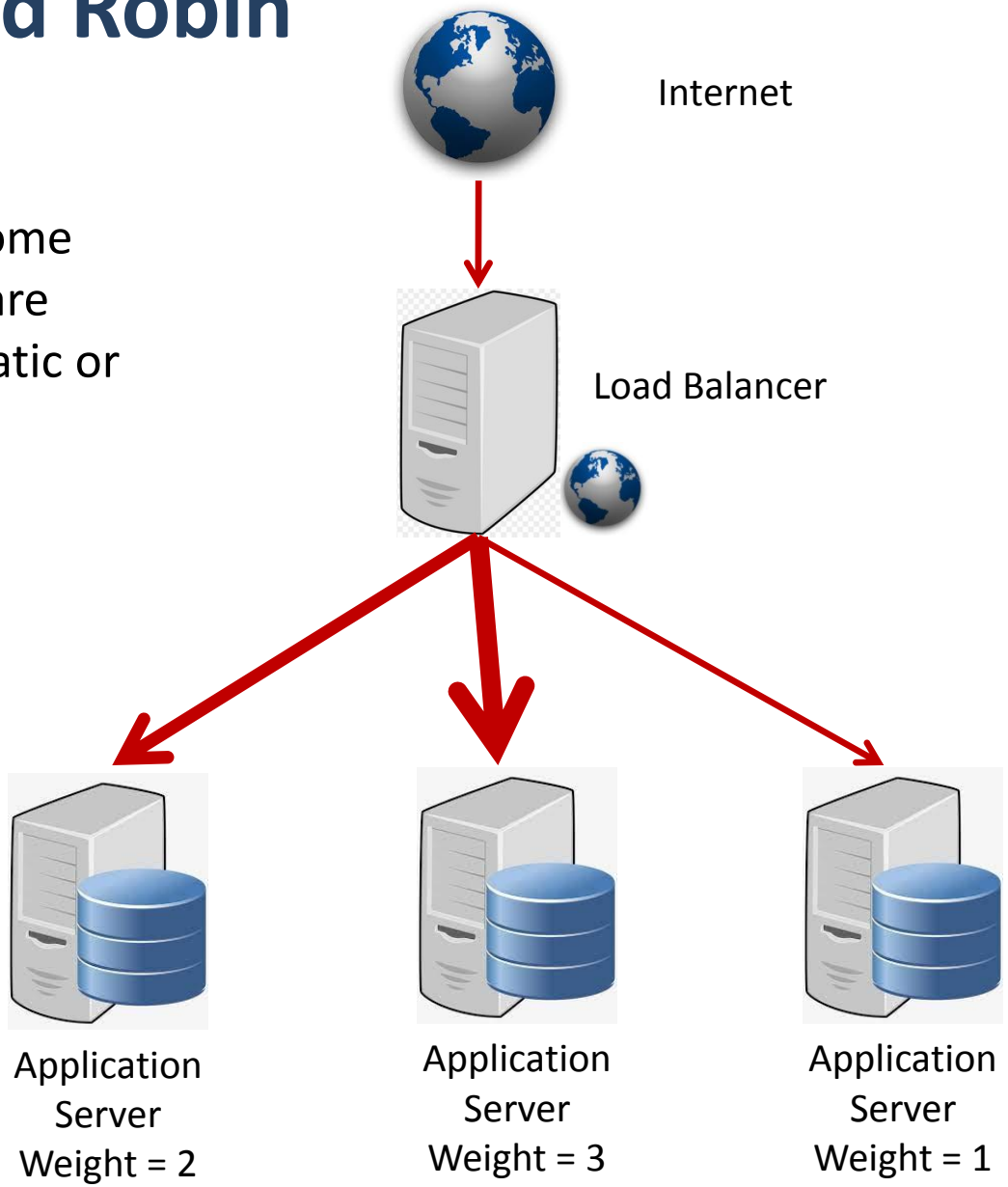
1. Round Robin

The servers are selected one by one in a non-hierarchical circular manner with no priority assigned to any specific server.



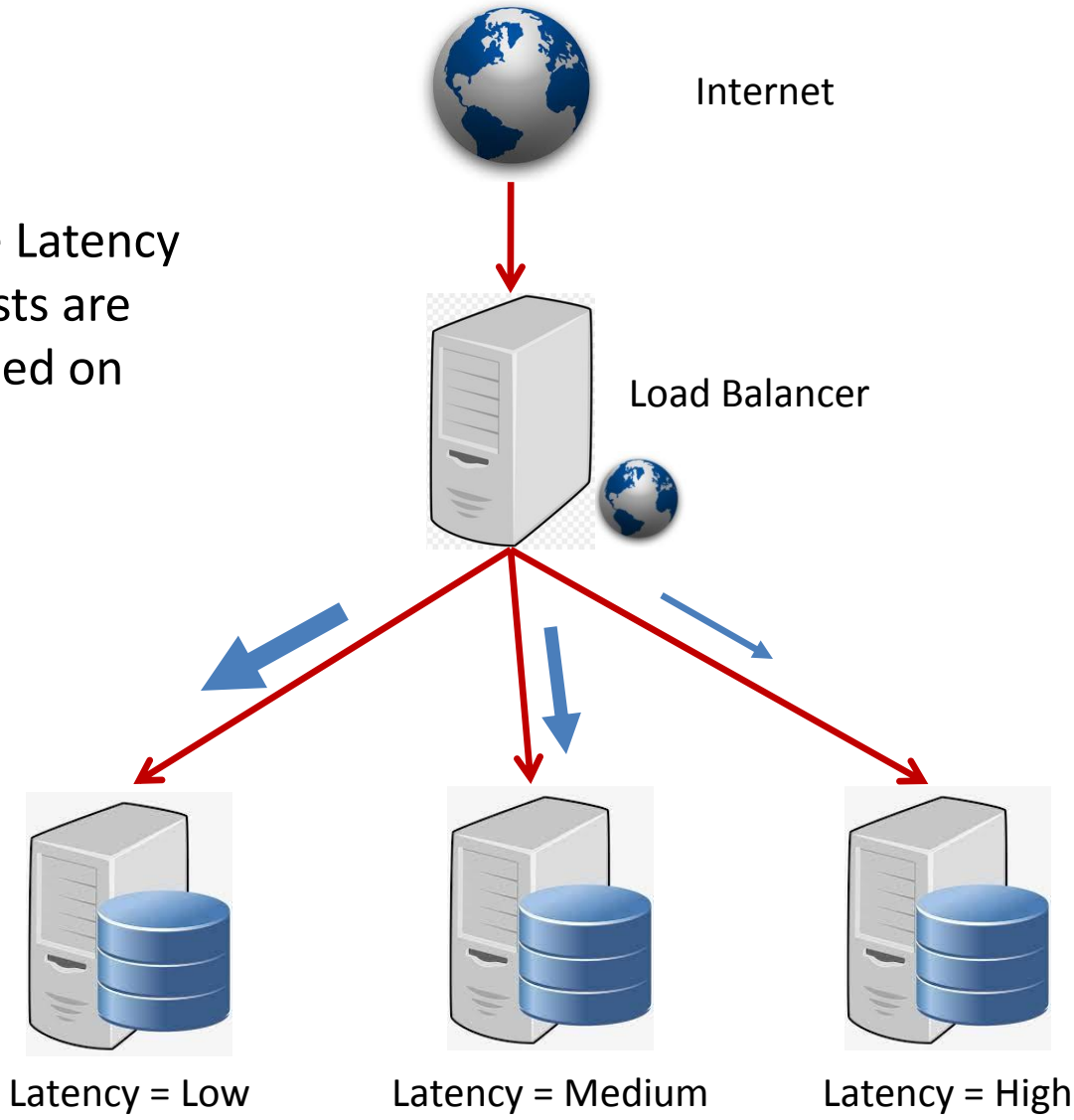
2. Weighted Round Robin

The servers are assigned with some weight. The incoming requests are routed proportionally using a static or dynamic ratio of their weights.



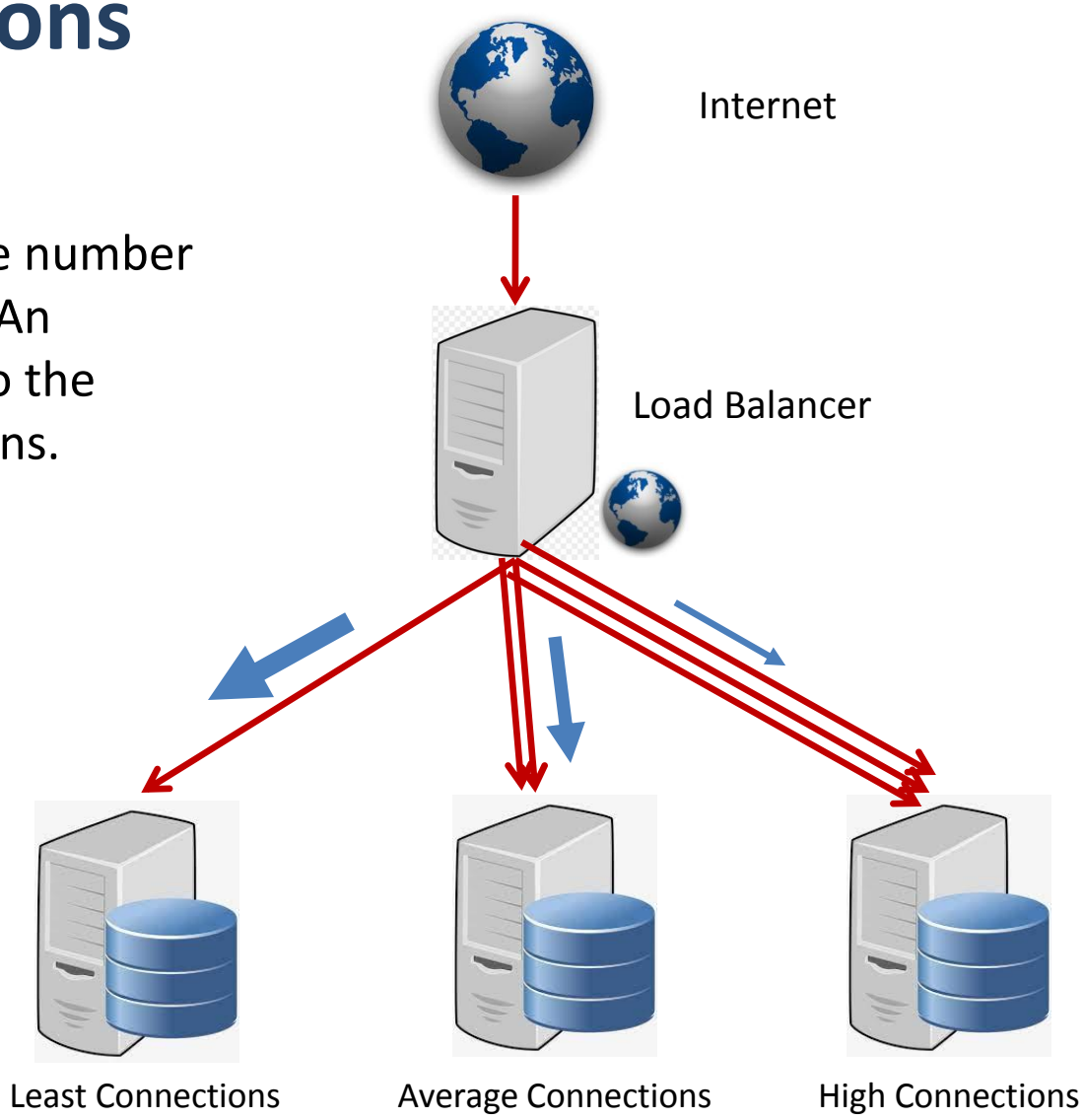
3. Low Latency

The load balancer monitors the Latency of each server. Incoming requests are proportionately distributed based on their latency.



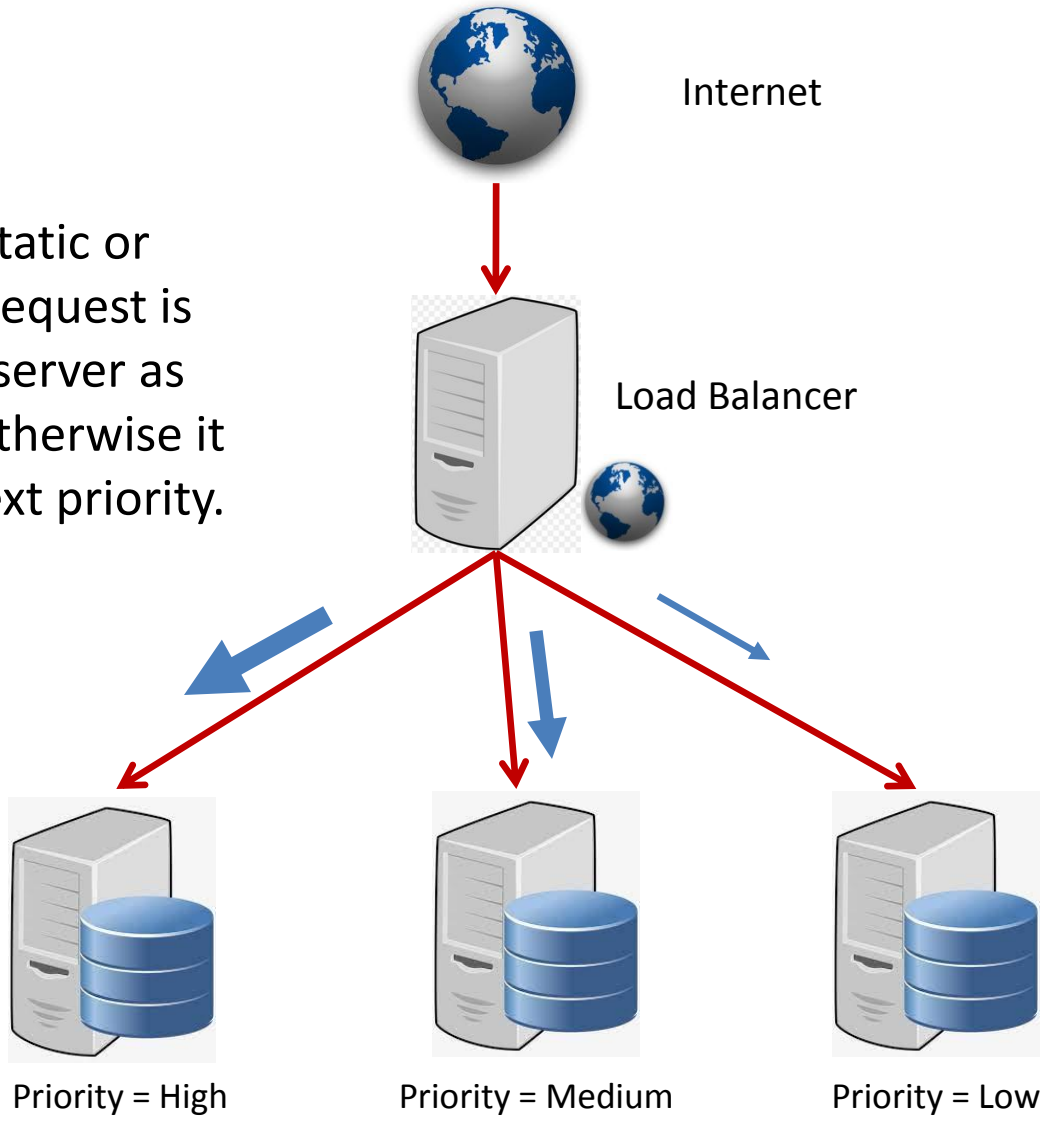
4. Least Connections

The load balancer monitors the number of connections to each server. An incoming request is assigned to the server that has least connections.



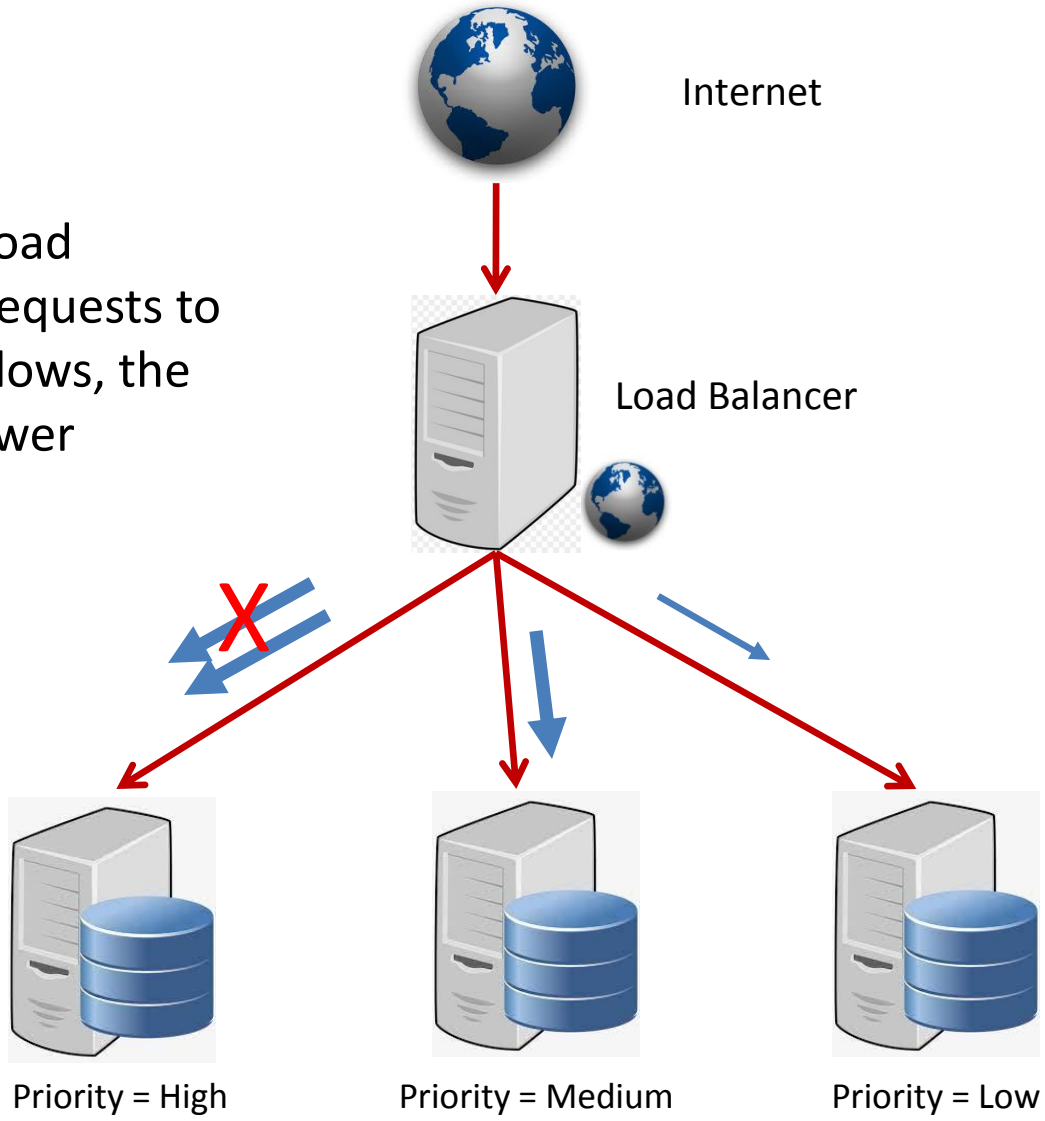
5. Priority

Each server is assigned with a (static or dynamic) priority. An incoming request is assigned to the highest priority server as long as the server is available; otherwise it is assigned to a server having next priority.



6. Overflow

This is similar to Priority based load balancing. When the incoming requests to the highest priority server overflows, the requests are assigned to next lower priority server.



Virtualization

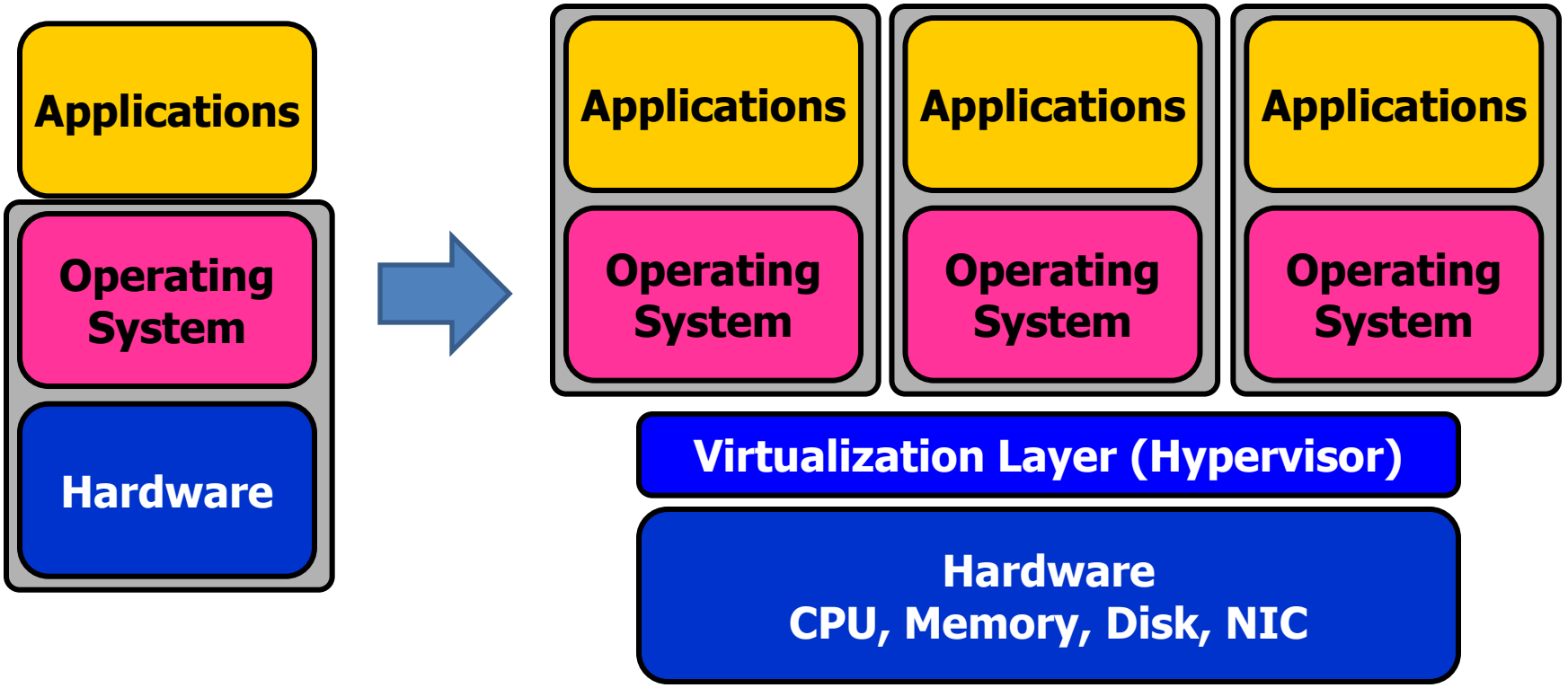
- Virtualization is one of the most important concepts in Cloud Computing
- It is the ability to run multiple operating systems on a single physical system and share the underlying hardware resources optimally.
- It is the process by which one computer hosts the appearance of many computers.
- Virtualization is used to improve IT throughput and costs by using physical resources as a pool from which virtual resources can be allocated.

Virtualization

Traditional Server Architecture

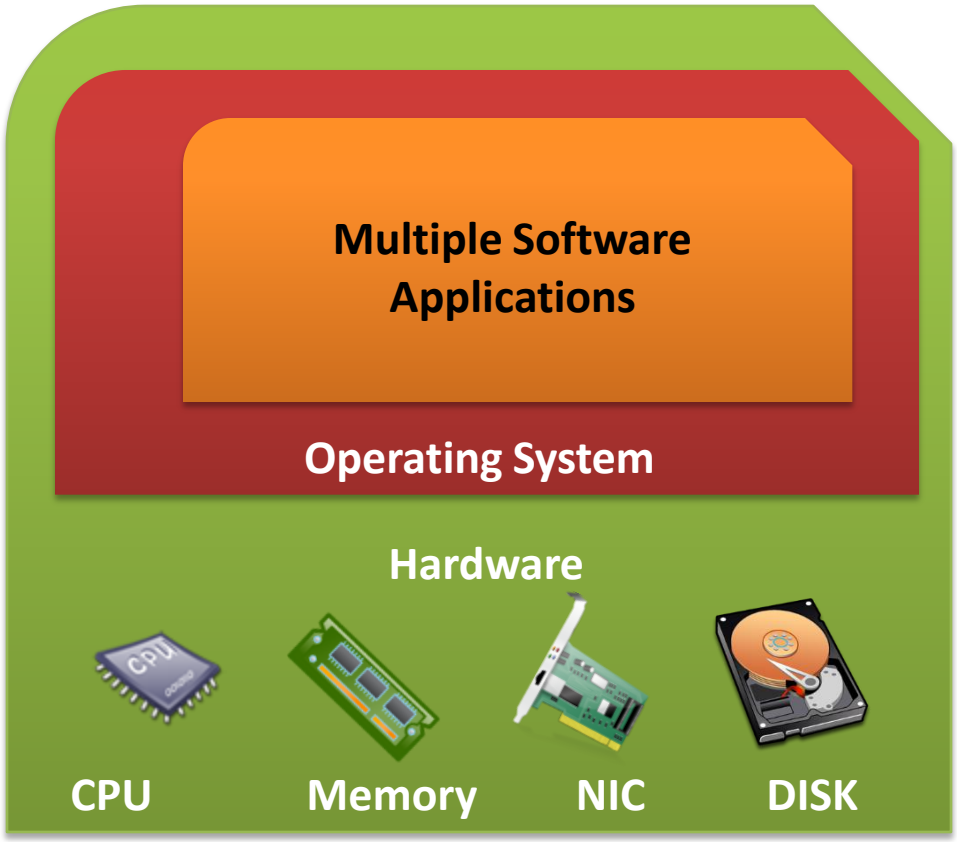
Virtualized Server Architecture

Virtualization



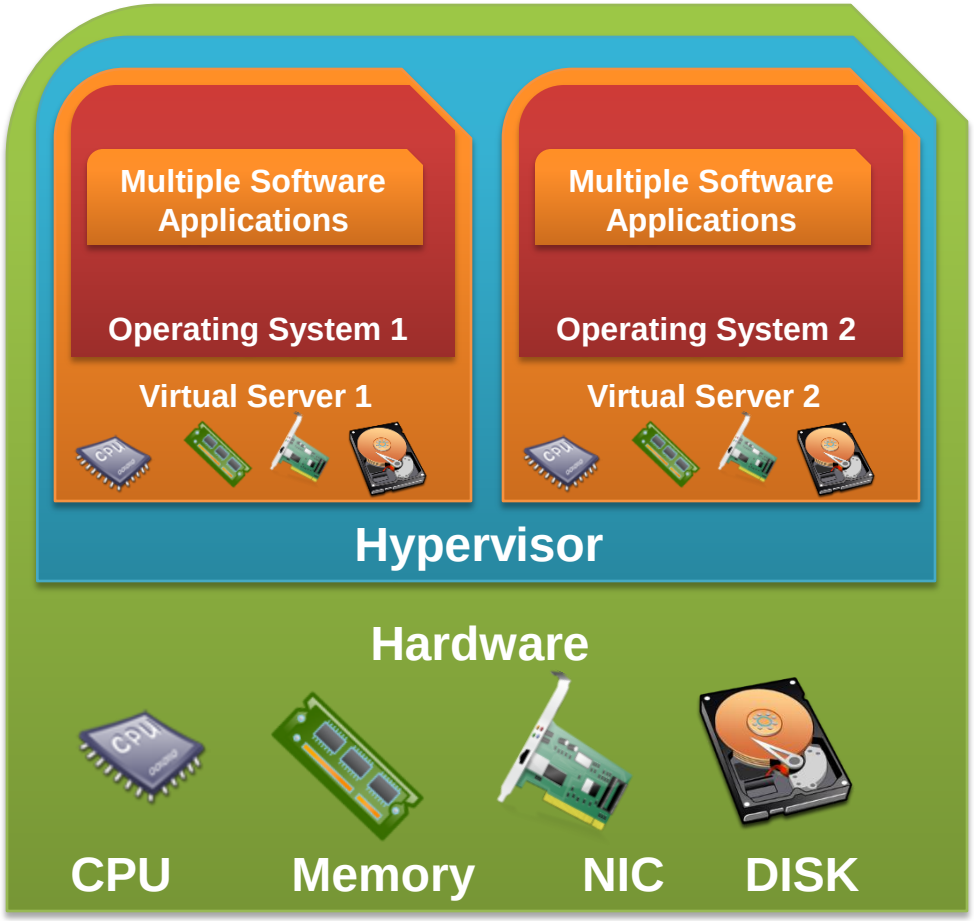
Hypervisor is a software program that manages multiple operating systems (or multiple instances of the same operating system) on a single computer system. The hypervisor manages the system's processor, memory, and other resources to allocate what each operating system requires. Hypervisors are designed for a particular processor architecture and are also called as **Virtual Machine Managers (VMM)**.

Server without Virtualization



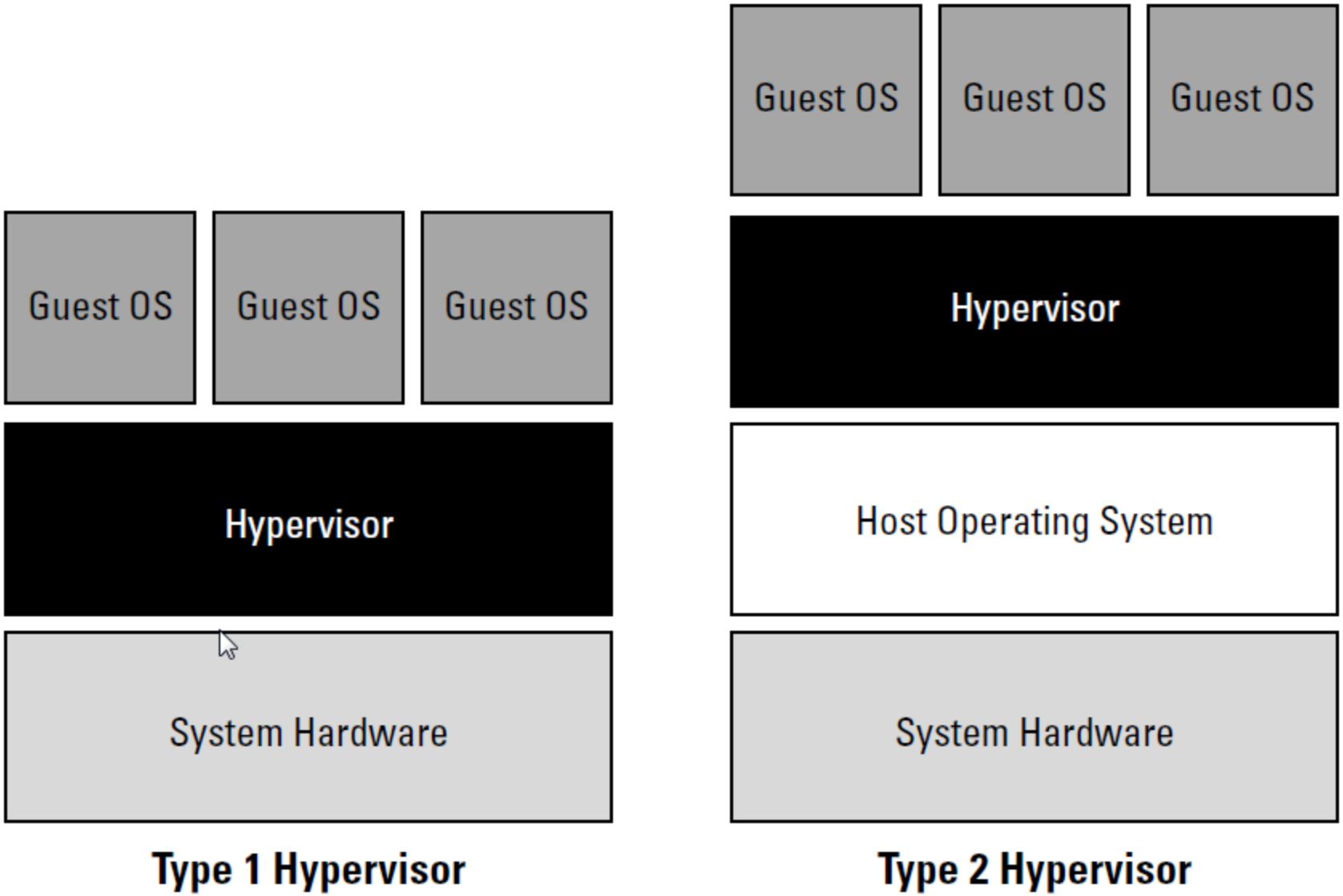
- Only one OS can run at a time within a server.
- Under-utilization of the resources.
- Inflexible and costly infrastructure.
- Hardware changes require manual effort and access to the physical server.

With Virtualization

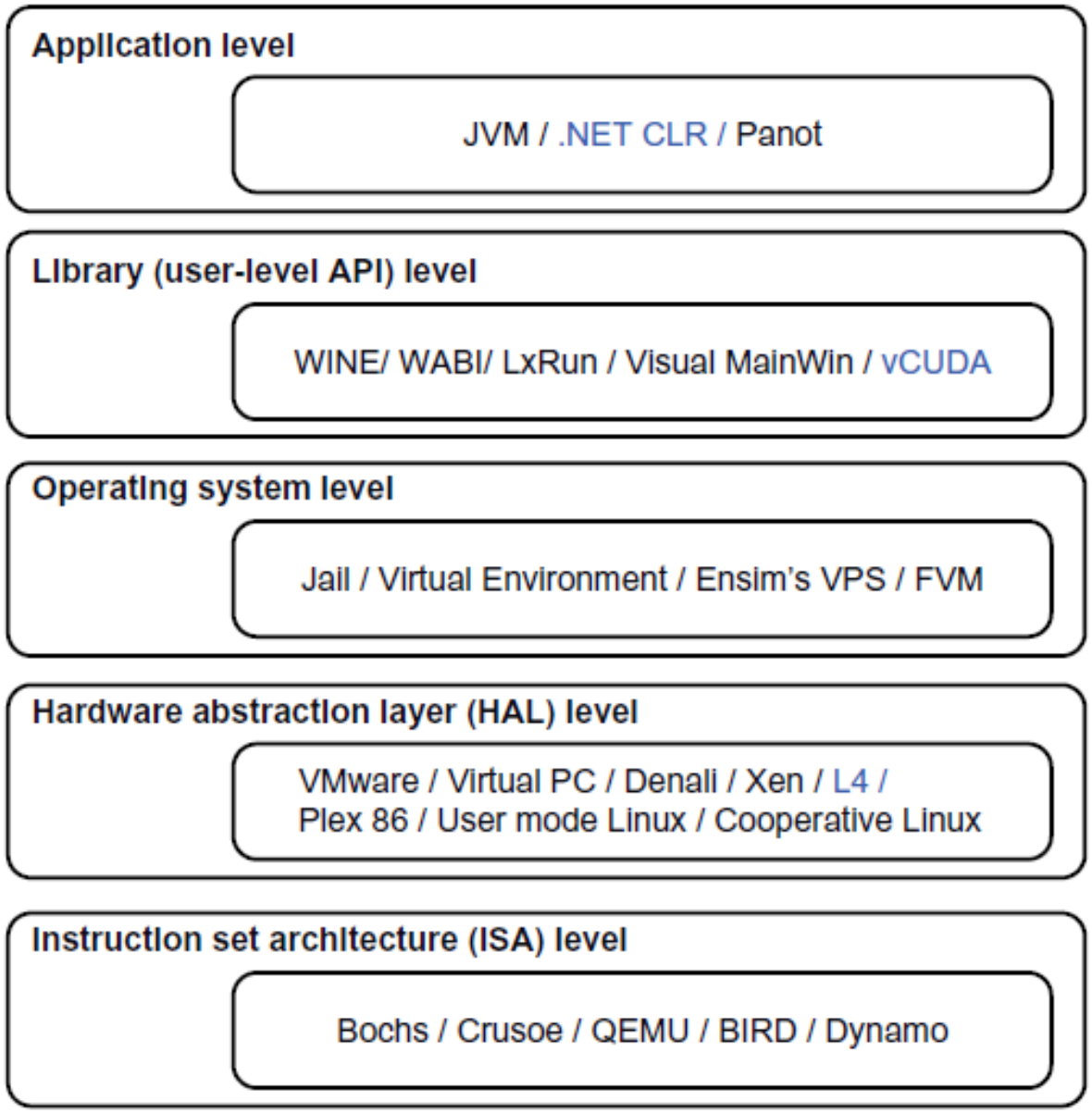


- Can run multiple OS simultaneously.
- Each OS can have different hardware configuration.
- Efficient utilization of hardware resources.
- Each virtual machine is independent.
- Easy to manage and monitor VMs centrally.
- Save electricity, cost to buy servers, space etc.

Types of Hypervisors



Levels of Virtualization



What can be virtualized?

1. Memory
2. CPU
3. Storage
4. Application
5. Desktop
6. Network
7. Hardware
8. Platform
9. … … …

Requirements of a VMM

- Defined by Popek & Goldberg (1974)
  1. Equivalence / Fidelity
     - A program running under the VMM should exhibit a behavior essentially identical to that demonstrated when running on an equivalent machine directly.
  2. Resource Control / Safety
     - The VMM must be in complete control of the virtualized resources.
  3. Efficiency / Performance
     - A statistically dominant fraction of machine instructions must be executed without VMM intervention.

Hypervisor Architecture

- Depending on Functionality two types:
  1. Micro-kernel architecture (ex. Microsoft Hyper-V)
     - Includes only basic functions like Memory Management and Processor Scheduling. Device drivers are outside the hypervisor
     - Code-base is smaller
  2. Monolithic Architecture (ex. VMWare ESX)
     - Includes all functions of micro-kernel architecture and also the functions of the device drivers
     - Code-base is larger

Types of Virtualization

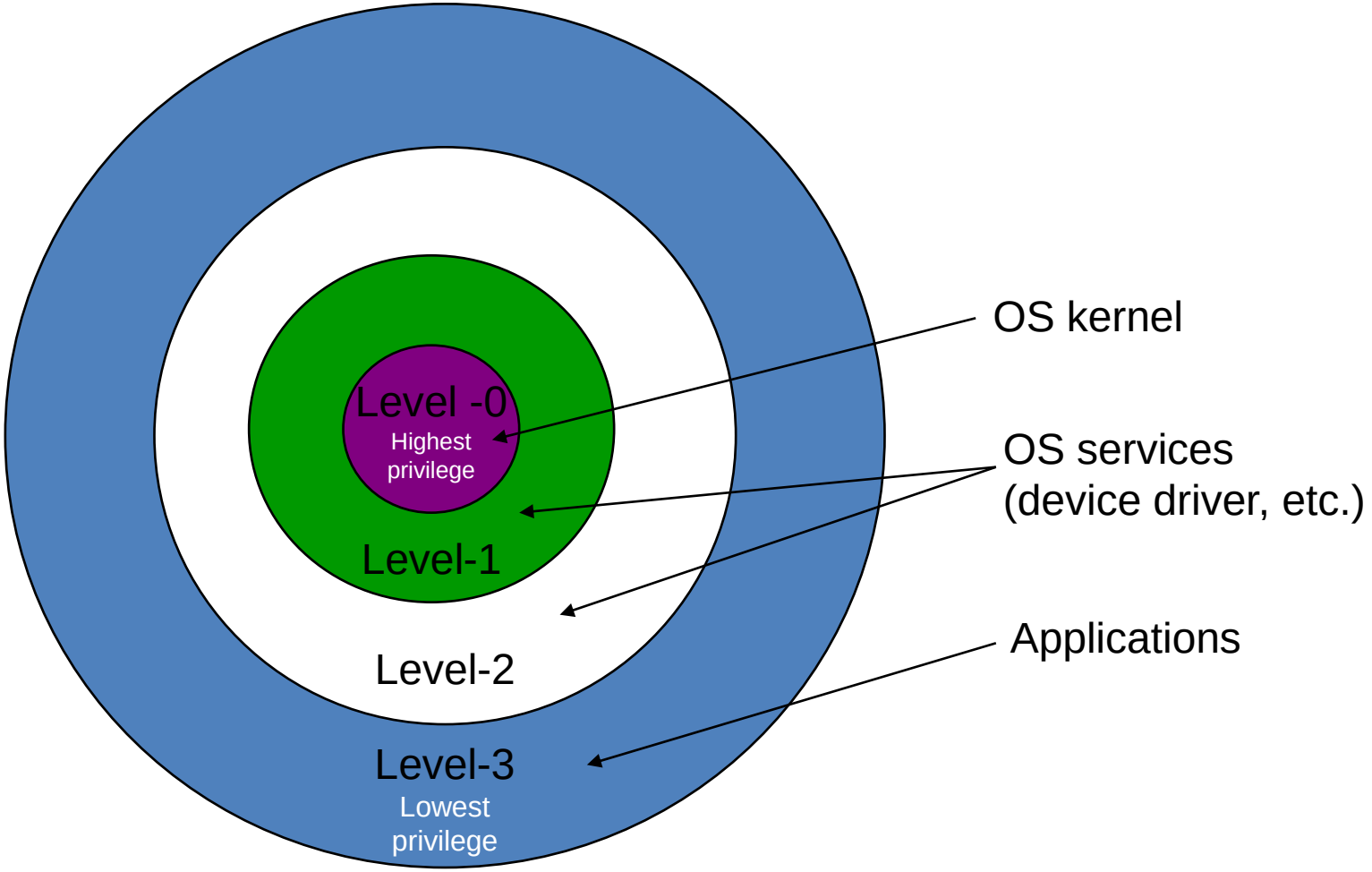
Different authors have classified virtualization into various different categories. We'll study the following techniques of virtualization:

- Full Virtualization
- Binary Translation
- Para-Virtualization (or OS-Assisted Virtualization)
- Hardware Assisted Virtualization
- OS-Level Virtualization
- Hosted Environment

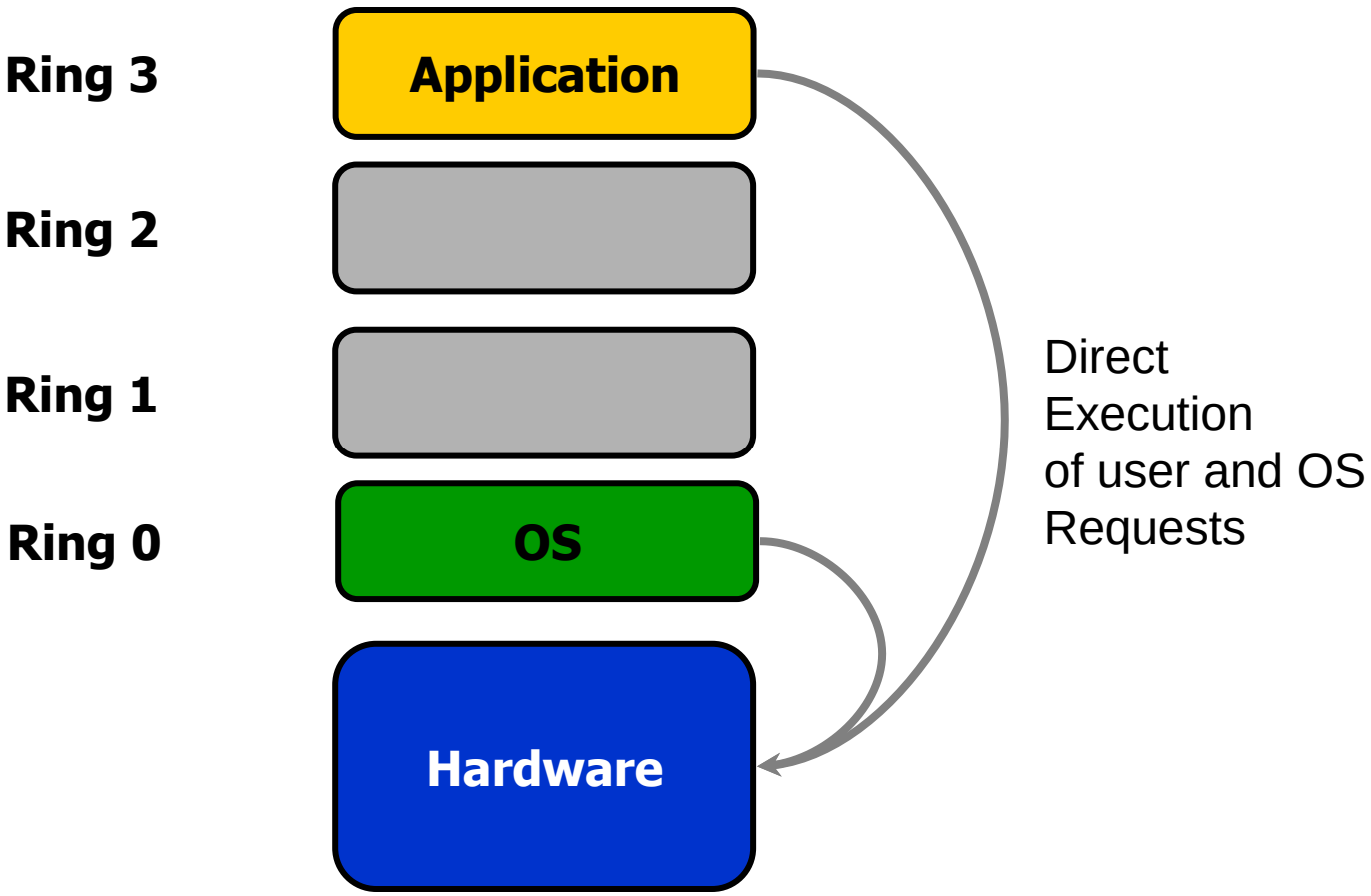
Full Virtualization

- A virtual machine environment that provides a complete simulation of the underlying hardware.
- All software (including all OS's) capable of execution on the raw hardware can be run in the virtual machine.
- Comprehensively simulate all computing elements as instruction set, main memory, interrupts, exceptions, and device access.
- Full virtualization is only possible given the right combination of hardware and software elements.
- Main Challenge:
  - Effects of every operation of a VM must be kept within the VM
  - Some machine instructions cannot be virtualized

Protection Levels of OS



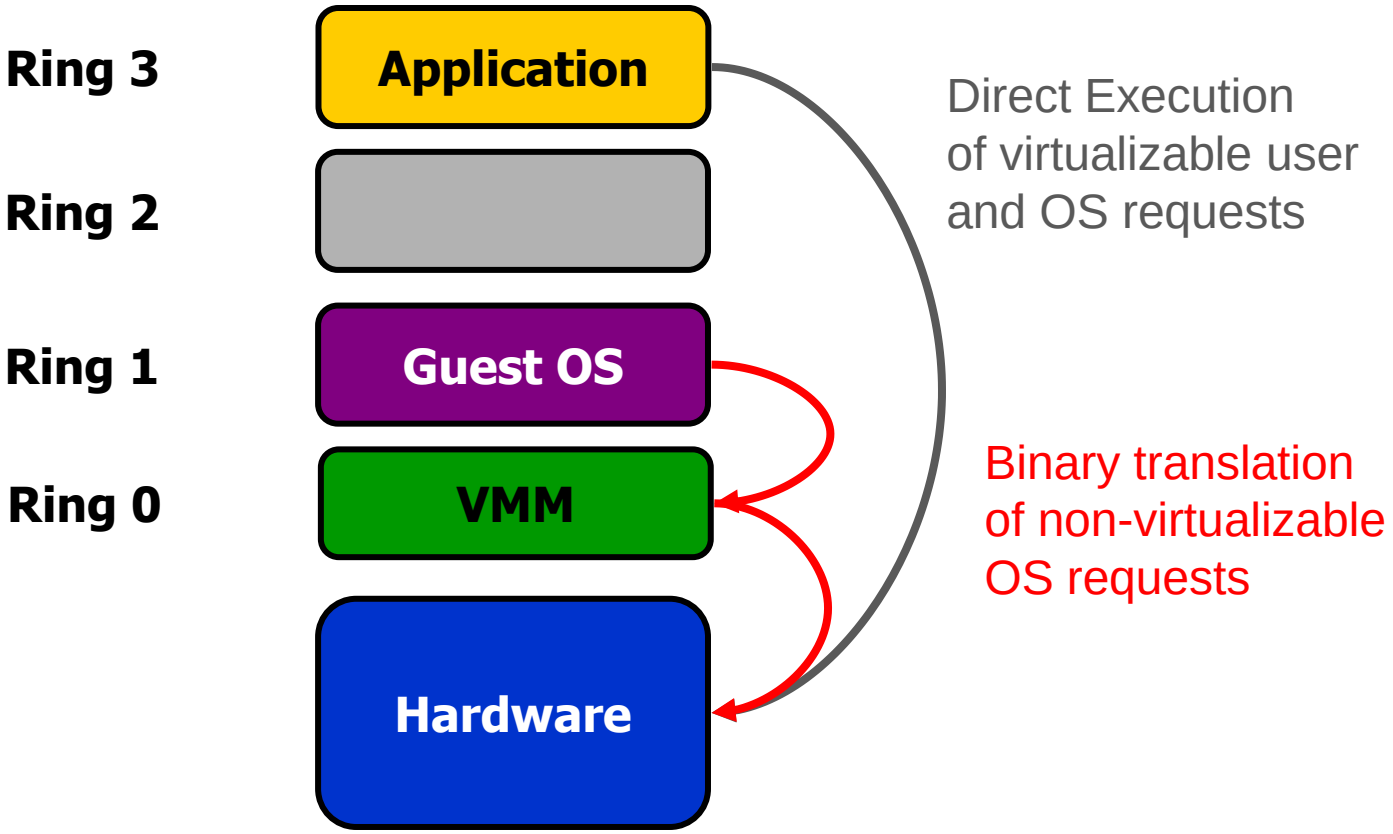
Protection Rings in x86 Architecture



Full Virtualization with Binary Translation

- Non-virtualizable instructions are translated with new instructions to produce the same effect
- Combination of binary translation and direct execution makes Full Virtualization possible
- The guest OS is fully abstracted (or decoupled) from the underlying hardware by the VMM layer
- The guest OS is NOT aware it is being virtualized and therefore require no modification.
- The hypervisor translates all sensitive instructions on-the-fly and caches the results for future use
- User level non-sensitive instructions run directly without any modification/translation at native speed.

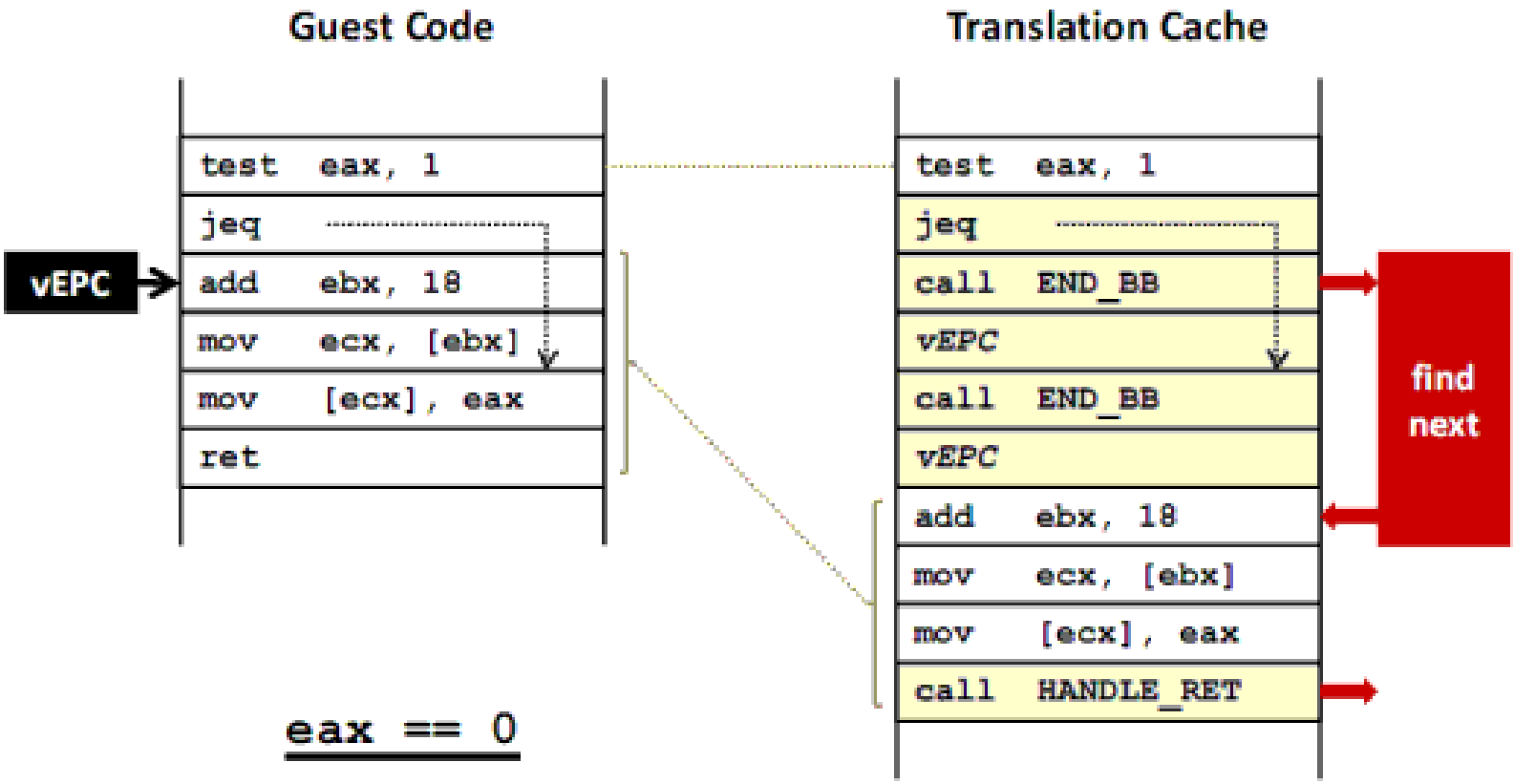
Binary Translation



The VMM puts itself at Ring 0 which is the highest privilege level

Examples: VMWare, Microsoft Virtual Server

Binary Translation



Binary translation from x86 to x86 virtualized in action. (Source: VMware)

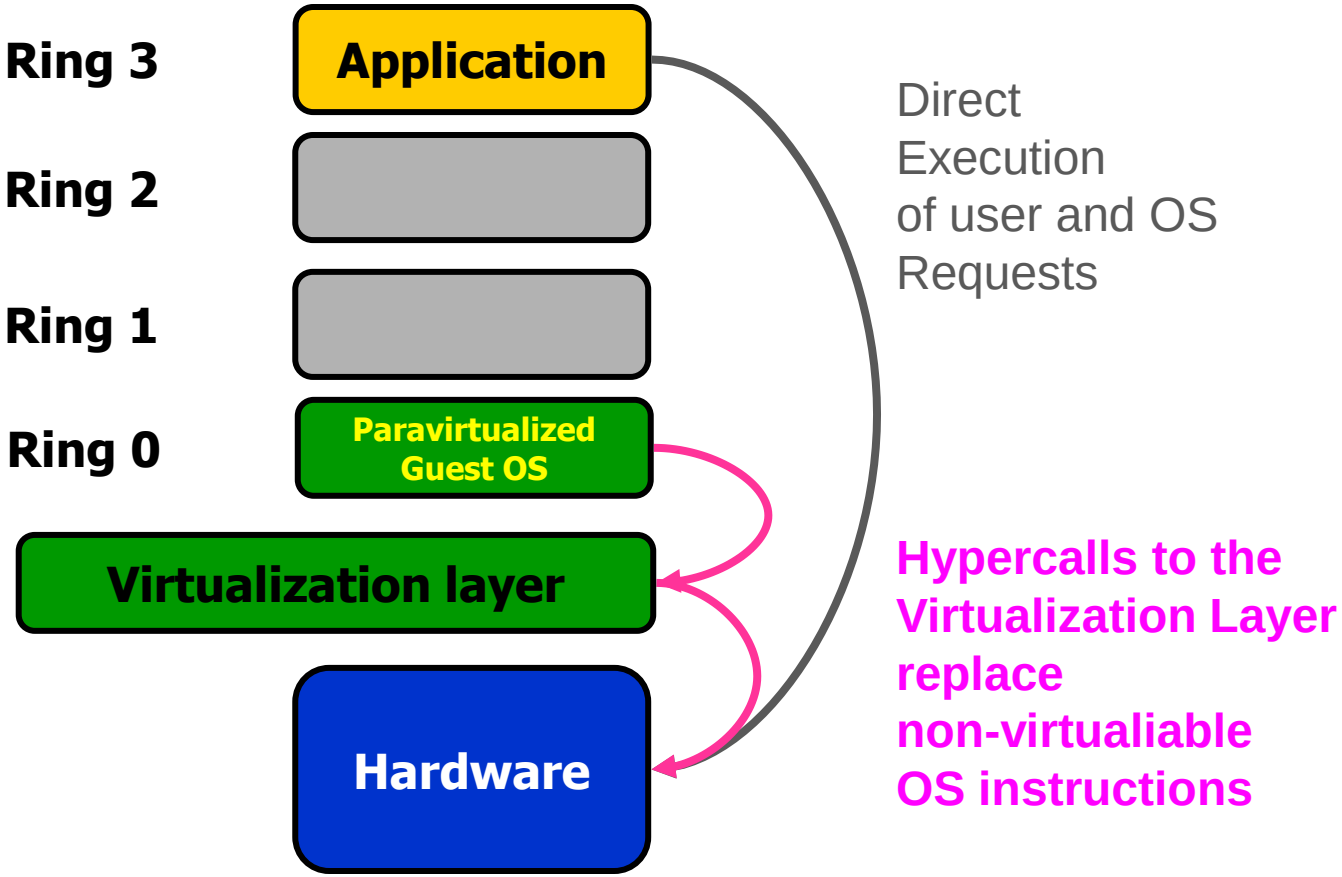
Binary Translation - *Disadvantages*

- Performance may not be ideal because binary translation can be time-consuming
- Full virtualization of I/O intensive applications could be a big challenge
- Caching of results of binary translation increases memory usage → Less memory is available for sharing between guest OS'es
- Performance is about 80-97% of host machine

Para-Virtualization (OS-Assisted)

- Binary translation is not easy as it seems!
- It is difficult to build more sophisticated/complex Binary Translation support for full virtualization
- Solution: - Modify the Guest OS → Para-Virtualization
- OS Kernel is modified to replace non-virtualizable instructions with *hypercalls* that communicate directly with the VMM (hypervisor).
- The hypervisor also provides *hypercall* interfaces (APIs) for other critical kernel operations such as memory management, interrupt handling and time keeping
- The guest OS is aware that they are being virtualized

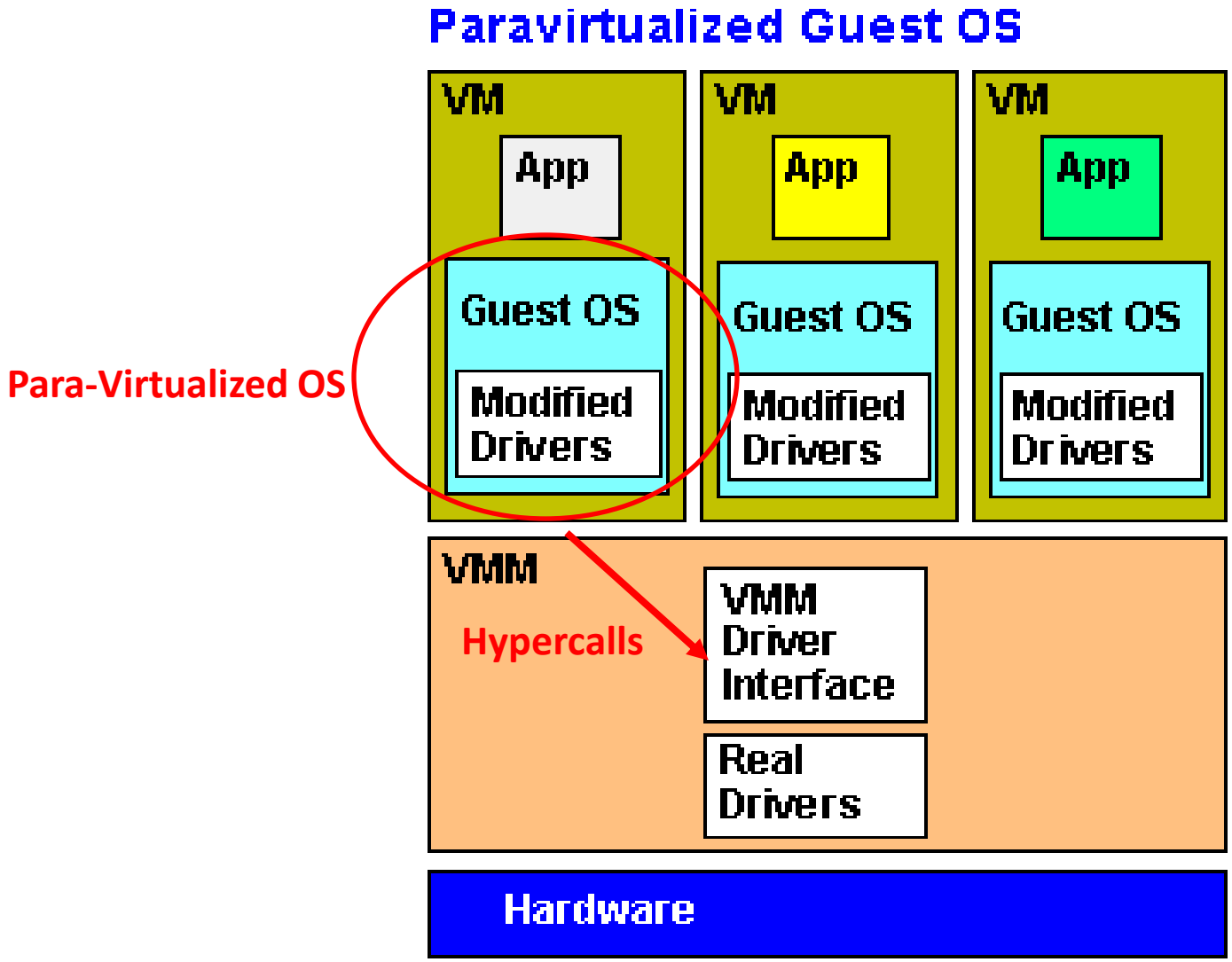
Para-Virtualization (OS-Assisted)



Advantage: Reduces the overhead of Binary Translation & Caching

Examples: Kernel-based VM (Linux), Xen, VMWare ESX

Para-Virtualization (OS-Assisted)



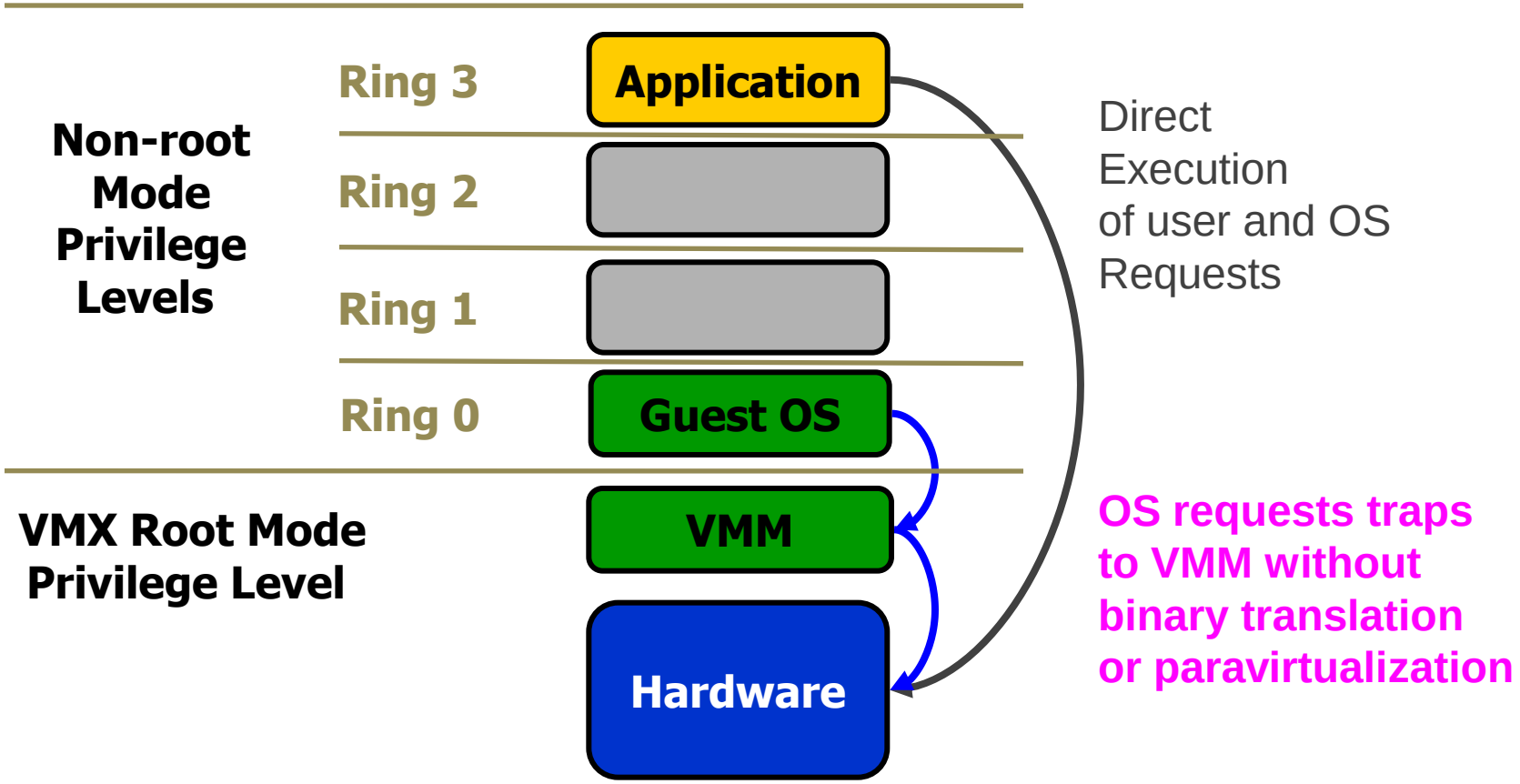
Para-Virtualization: *Disadvantages*

- Para-Virtualization does NOT support unmodified OS
- Guest OS's Kernel is modified for virtualization
  - it can no longer run on hardware directly
- Compatibility & portability is an issue
- Cost of maintaining para-virtualized OS is high
  - because they require deep kernel modifications
- Performance advantage may vary greatly depending on workload variations

Hardware-Assisted Virtualization

- This is also known as Accelerated Virtualization, Hardware Virtual Machine (e.g. Xen), or Native Virtualization (e.g. Virtual Iron)
- Hardware design of the CPU for privileged instructions with a new CPU execution mode -- allows the VMM to run in a root mode below Ring 0 – called “VMX root” mode.
  - Examples: Intel VT-x & VT-I Processors, AMD ‘s AMD-v Processors
- Privileged and sensitive calls are set to automatically trap to the hypervisor, removing the need for either Binary Translation or Para-Virtualization.
- The guest state is stored in Virtual Machine Control Structures (VT-x) or Virtual Machine Control Blocks (AMD-V).

Hardware-Assisted Virtualization



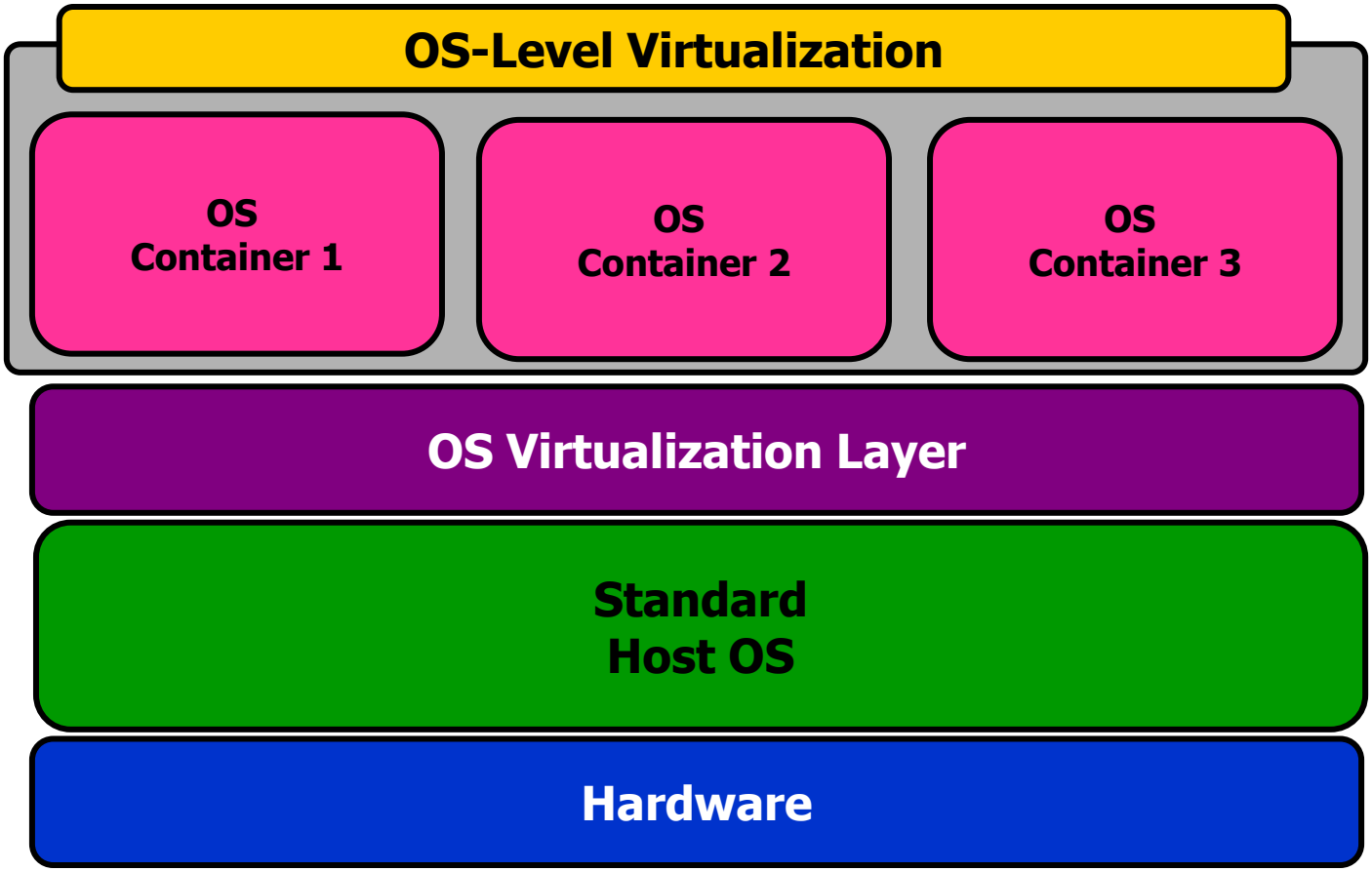
Hardware-Assisted Virtualization

- Advantages:
  - Reduces /eliminates the maintenance overhead of para-virtualization as it does not need changes in the guest operating system
  - It is considerably easier to obtain better performance
- Disadvantages:
  - Requires explicit support in the host CPU
  - Involves many VM traps → high CPU overheads
  - Limited scalability & efficiency
  - Rigid programming model

OS-Level Virtualization

- Kernel of a host OS allows for multiple isolated user-space instances or Containers. Each guest OS instance is placed inside a container and looks & feels like a real server.
- Works at the operating system (kernel) layer
- The isolated containers on a single physical server contains an OS instance to utilize hardware, software with maximum efficiency.
- OS-level virtualization implementations are capable of live migration can be used for dynamic load balancing of containers between nodes in a cluster.

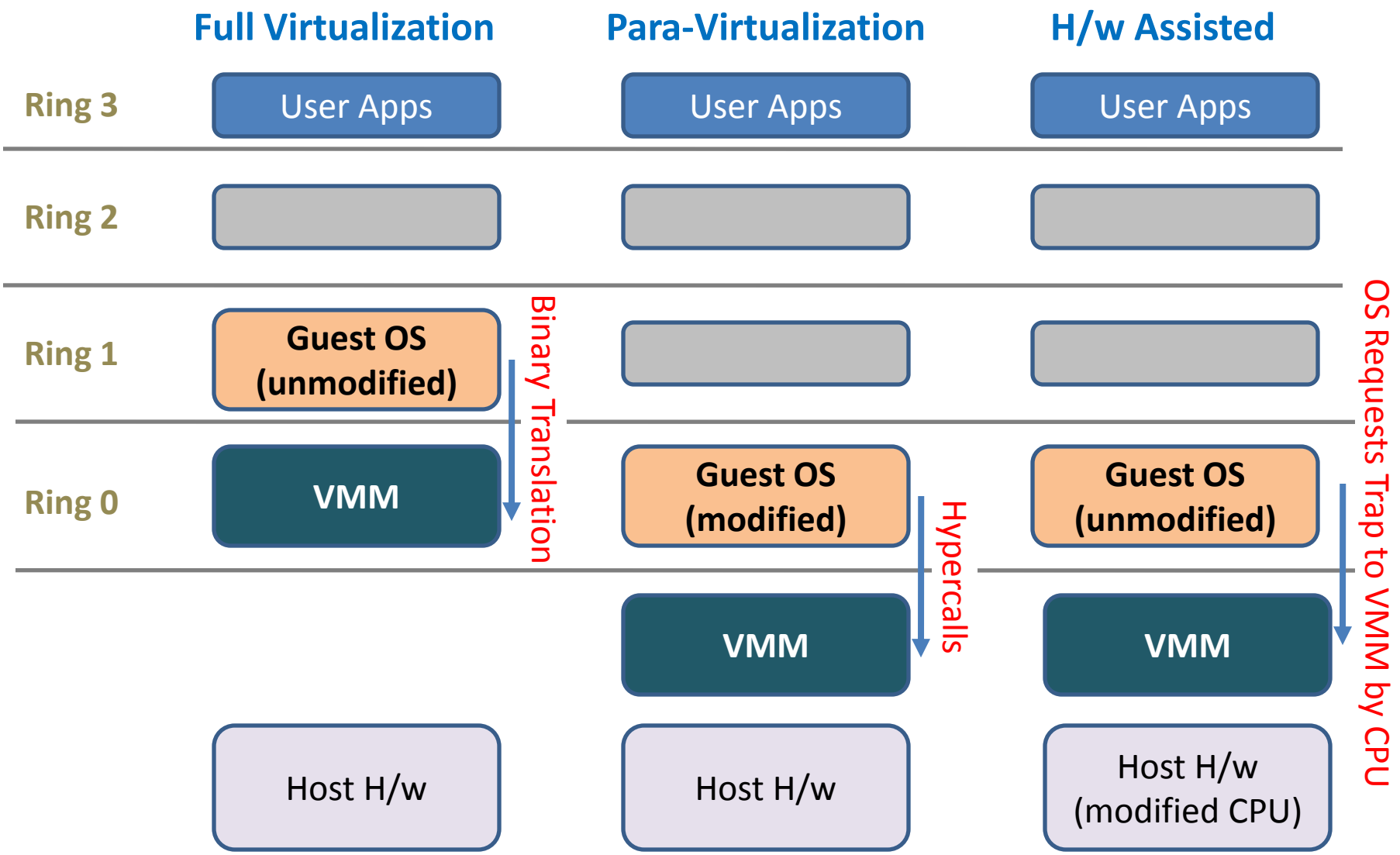
OS-Level Virtualization



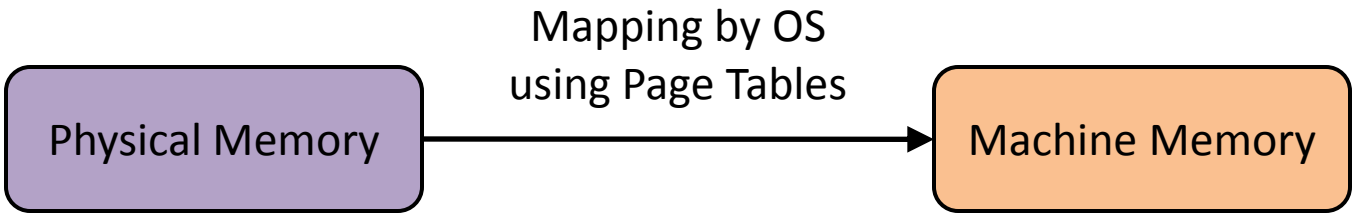
Hosted Environment (User-mode Linux)

- **UML** enables multiple virtual Linux kernel-based OS'es (known as guests) to run as applications within a normal Linux system (known as the host).
- Each guest OS is just like a normal application running as a process in user space
- This approach provides the user with a way of running multiple virtual Linux machines on a single piece of hardware
- It offers some isolation, generally without affecting the host environment's configuration or stability.

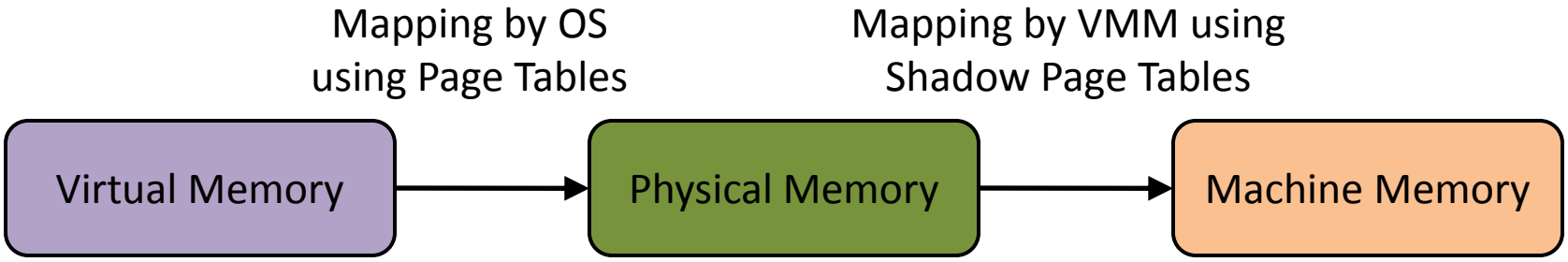
Comparison



Memory Virtualization



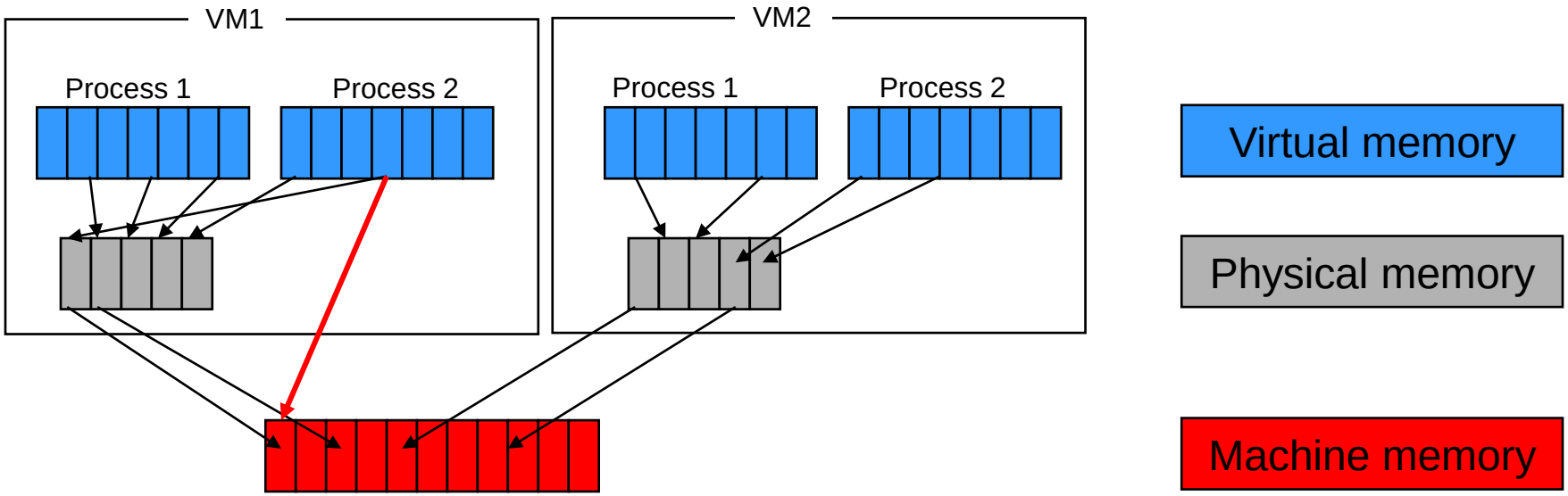
Traditional Execution Environment



Virtual Execution Environment

Memory Virtualization

- To run multiple virtual machines on a single system, another level of memory virtualization is required (two-stage mapping)
- The VMM is responsible for mapping guest physical memory to the actual machine memory, and it uses shadow page tables to accelerate the mappings



Cloud Implementations

IaaS

- Amazon Web Services
- Elastic Compute Cloud (EC2)
- Simple Storage Service (S3)
- Simple Queuing Service (SQS)
- VMWare vCloud, vCloud Express

PaaS

- Google App Engine
- Java & Python Runtime Environments
- Windows Azure
- SQL Azure
- Windows Azure AppFabric

SaaS

- SalesForce.com,
- Force.com, Force Database
- MS Office Live
- LiveMesh.com
- Google Apps

Amazon Web Services (AWS)

- Amazon Web Services (AWS) is a collection of remote computing services (web services) that together make up a cloud computing platform, offered over the Internet by Amazon.com.
- Website: http://aws.amazon.com
- AWS is located in **18 Geographical Regions** and 1 Local Region
- Each Region is wholly contained within a single country and all of its data and services stay within the designated Region.
- Each Region has multiple 'Availability Zones', which are distinct data centers providing AWS services (55 Availability Zones)
- Availability Zones are isolated from each other to prevent outages from spreading between Zones.
- However, several services also operate across Availability Zones
  - Example: S3, DynamoDB etc.

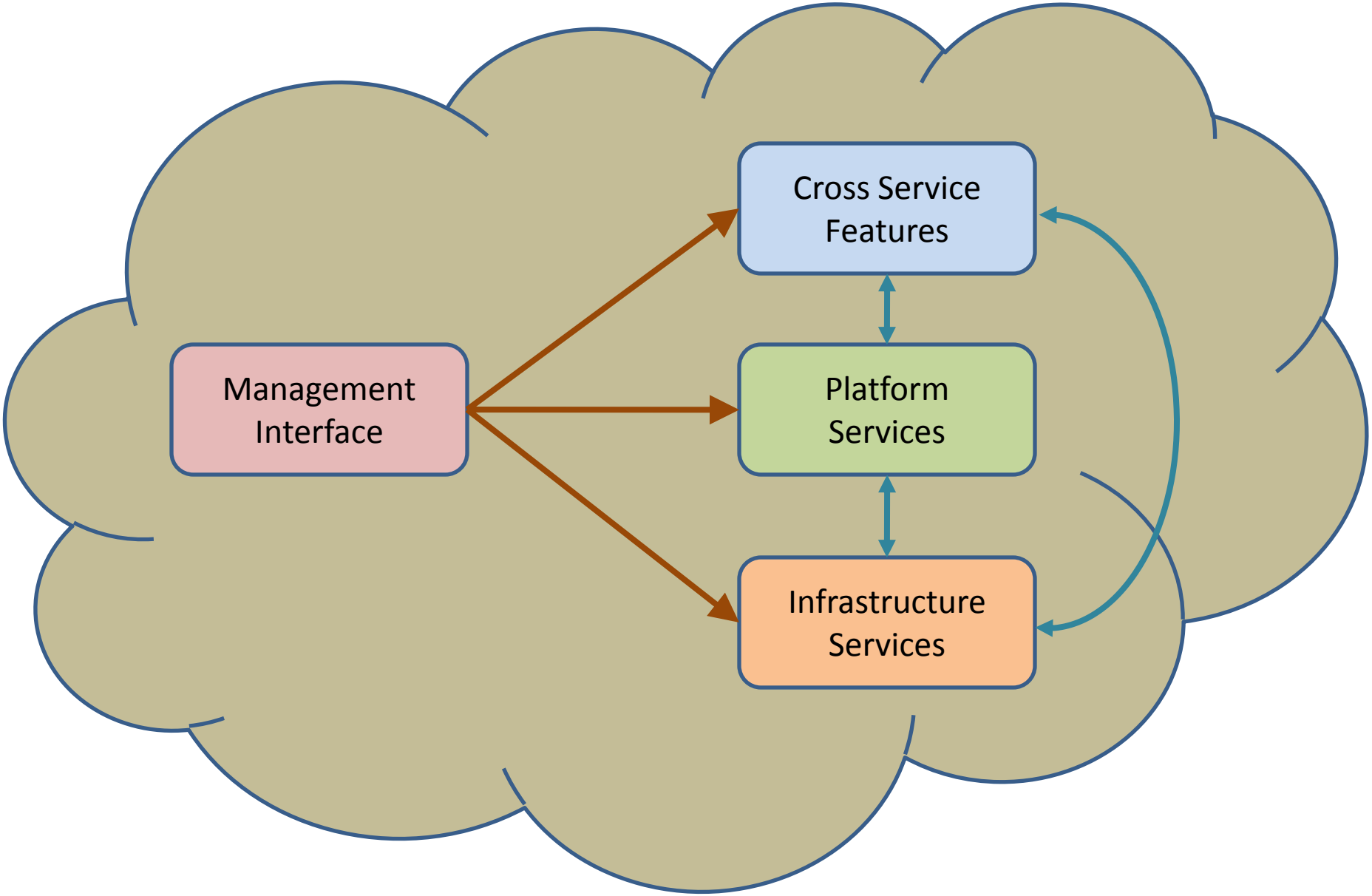
AWS Offerings

- **Low Ongoing Cost**: pay-as-you-go pricing with no up-front expenses or long-term commitments.
- **Instant Elasticity & Flexible Capacity**: (scaling up and down) Eliminate guessing on your infrastructure capacity needs.
- **Speed & Agility**: Develop and deploy applications faster Instead of waiting weeks or months for hardware to arrive and get installed.
- **Apps not Ops**: Focus on projects. Lets you shift resources away from data center investments and operations and move them to innovative new projects.
- **Global Reach**: Take your apps global in minutes.
- **Open and Flexible**: You choose the development platform or programming model that makes the most sense for your business.
- **Secure**: Allows your application to take advantage of the multiple layers of operational and physical security in the AWS data centers to ensure the integrity and safety of your data.

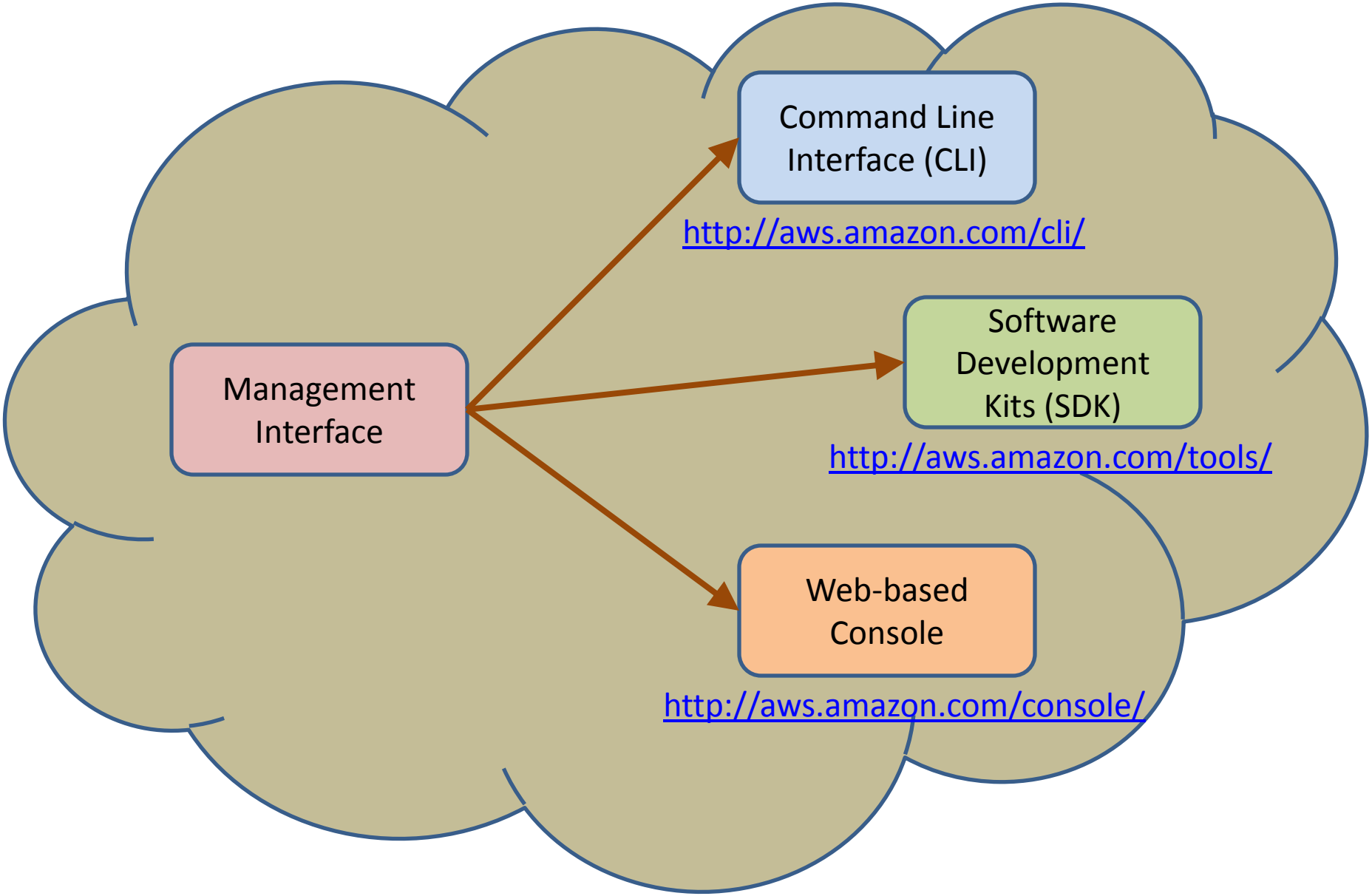
Amazon Web Services (AWS)

- Compute
  - Elastic Compute Service (EC2)
  - Elastic MapReduce
  - Auto Scaling
- Storage
  - Simple Storage Service (S3)
  - Elastic Block Store (EBS)
  - AWS Import/Export
- Messaging
  - Simple Queue Service (SQS)
  - Simple Notification Service (SNS)
- Database
  - SimpleDB
  - Relational Database Service (RDS)
- Content Delivery
  - CloudFront
- Networking
  - Elastic Load Balancing
  - Virtual Private Cloud
- Monitoring
  - CloudWatch
- Workforce
  - Mechanical Turk

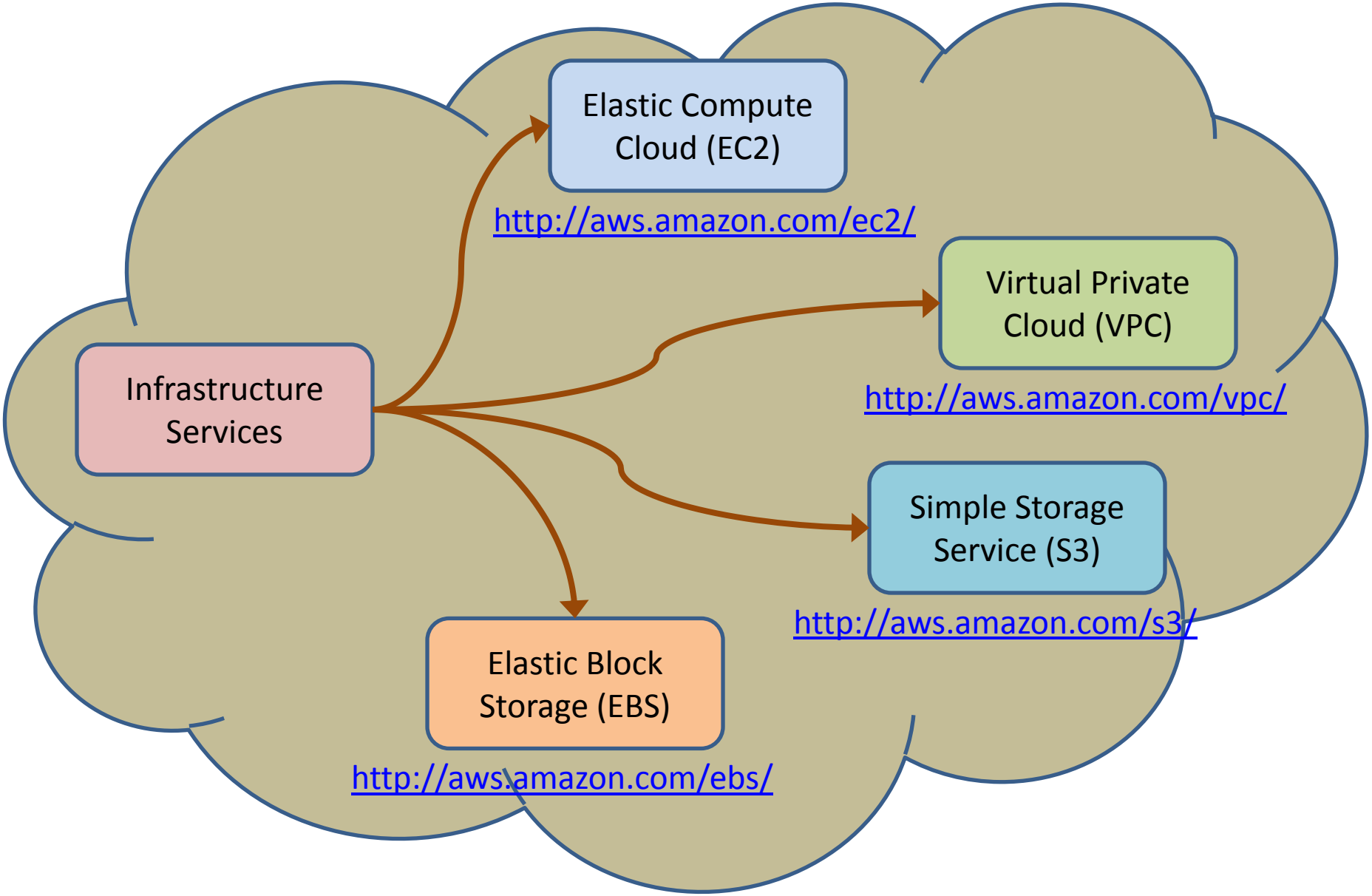
The AWS Universe



AWS Management Interface



AWS Infrastructure Services



Amazon Elastic Compute Cloud (EC2)

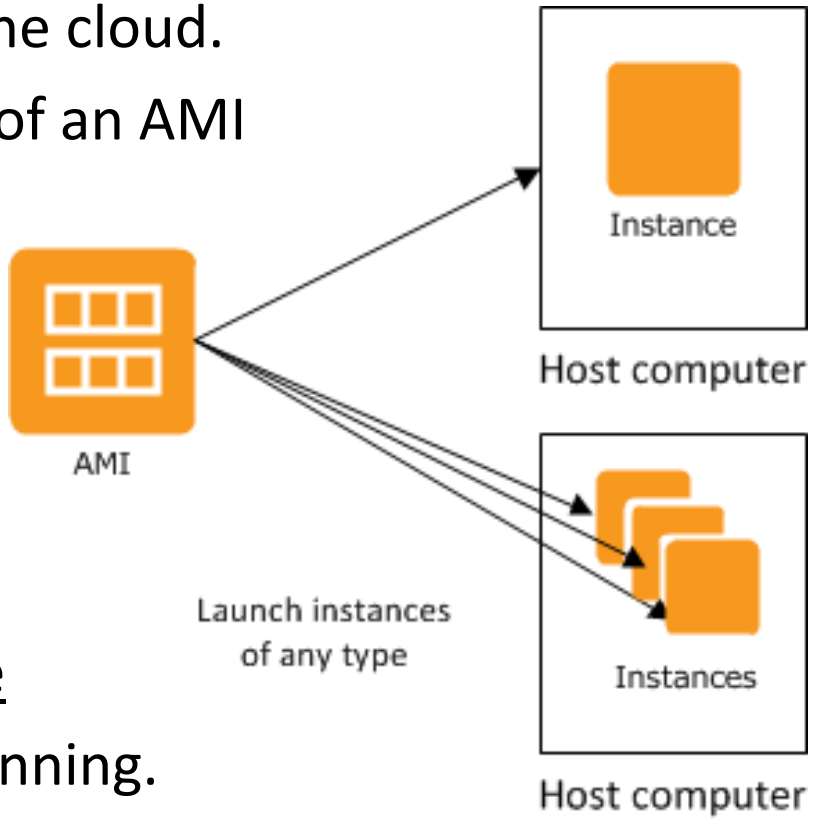
- An Amazon Web Service that provides resizable compute capacity in the cloud.
- Customers can access servers, software, and storage resources over the Internet in a self-service manner
- EC2 allows creating Virtual Machines (VM) on-demand. Pre-configured templated Amazon Machine Image (AMI) can be used get running immediately.
- Creating and sharing your own AMI is also possible via the AWS Marketplace.
- Auto Scaling allows seamless dynamic scaling (up/down) depending on demand to maintain performance and minimize costs.

Amazon Elastic Compute Cloud (EC2)

- Elastic load balancing automatically distributes incoming application traffic across multiple Amazon EC2 instances.
- Provide tools to build failure resilient applications by launching application instances in separate Availability Zones.
- Pay only for resources actually consume – computed and charged by instance-hours.
- VM Import/Export enables you to easily import virtual machine images from your existing environment to Amazon EC2 instances and export them back at any time.

Instances & AMIs

- An *Amazon Machine Image (AMI)* is a template that contains a software configuration (for example, an operating system, an application server, and applications).
- From an AMI, one can launch an *instance*, which is a copy of the AMI running as a virtual server in the cloud.
- One can launch multiple instances of an AMI
- The instances keep running until they are stopped, terminated or until they fail.
- A launched instance looks like a traditional host with root privilege.
- The AWS account has a limit on the number of instances that can be running.



EC2 Instance Types

- There are 5 categories of EC2 Instances
  - General Purpose
  - Compute Optimized
  - Memory Optimized
  - Accelerated Computing
  - Storage Optimized
- Under each category, there are several instance types:
  - nano, micro, small, medium, large, xlarge, 2xlarge
- Instances are named like T1, T3, M4, M5, C5 … …
- Some of the instance types are *burstable performance* instances.
- AWS Virtualization Types: PV or HVM

EC2 Instance Types

- General Purpose:
  - General purpose instances provide a balance of compute, memory and networking resources, and can be used for a variety of diverse workloads. These instances are ideal for applications that use these resources in equal proportions such as web servers and code repositories.
- Compute Optimized:
  - Compute Optimized instances are ideal for compute bound applications that need high performance processors.
  - Well suited for batch processing workloads, media transcoding, high performance web servers, high performance computing (HPC), scientific modeling, dedicated gaming servers and ad server engines, machine learning inference and other compute intensive applications.

EC2 Instance Types

- Memory Optimized:
  - These instances are designed to deliver fast performance for workloads that process large data sets in memory.
  - Well suited for memory intensive applications such as high performance databases, distributed in-memory caches, mid-size in-memory databases, real time big data analytics, etc.
- Accelerated Computing:
  - Use hardware accelerators, or co-processors, to perform functions, such as floating point number calculations, graphics processing, or data pattern matching, more efficiently than is possible in software running on CPUs.
  - Well suited for machine learning, deep learning, computational finance, seismic analysis, speech recognition, autonomous vehicles, drug discovery, etc.

EC2 Instance Types

- Storage Optimized:
  - Instances are designed for workloads that require high, sequential read and write access to very large data sets on local storage.
  - Optimized to deliver tens of thousands of low-latency, random I/O operations per second (IOPS) to applications.
  - Suitable for NoSQL databases (e.g. Cassandra, MongoDB, Redis), in-memory databases (e.g. Aerospike), scale-out transactional databases, data warehousing, ElasticSearch, analytics workloads.

➢ *Burstable Performance* Instances provide a baseline level of CPU performance with the ability to burst above the baseline depending on increase in demand.

AWS EC2 CPU Credits

- One CPU credit is equal to one vCPU running at 100% utilization for one minute
  - or, one vCPU running at 50% utilization for two minutes
  - or, two vCPUs running at 25% utilization for two minutes
- Each burstable performance instance continuously earns (per ms) a set rate of CPU credits per hour, depending on the instance size.
- If a burstable performance instance uses fewer CPU resources than is required for baseline performance (e.g. when it is idle), the unspent CPU credits are accrued in the CPU credit balance.

Amazon Elastic Block Store (EBS)

- Provides block level storage volumes (1 GB to 1 TB) for use with Amazon EC2 instances.
  - Multiple volumes can be mounted to the same instance.
  - EBS volumes are network-attached, and persist independently from the life of an instance.
  - Storage volumes behave like raw, unformatted block devices, allowing users to create a file system on top of Amazon EBS volumes, or use them in any other way a block device (like a hard drive) is used.
- EBS volumes are placed in a specific Availability Zone,
- The EBS volumes can then be attached to EC2 instances also in that same Availability Zone.

Amazon Elastic Block Store (EBS)

- Each storage volume is automatically replicated within the same Availability Zone.
- EBS provides the ability to create point-in-time snapshots of volumes, which are persisted to Amazon S3.
- These snapshots can be used as the starting point for new Amazon EBS volumes, and protect data for long-term durability.
- The same snapshot can be used to instantiate as many volumes as needed.
- These snapshots can be copied across AWS Regions.

EBS Volumes

- Standard volumes offer storage for applications with moderate or bursty I/O requirements.
  - Standard volumes deliver approximately 100 IOPS on average.
  - Well suited for use as boot volumes, where the burst capability provides fast instance start-up times.
- Provisioned IOPS volumes are designed to deliver predictable, high performance for I/O intensive workloads such as databases.
  - Customer can specify an IOPS rate when creating a volume, and EBS provisions that rate for the lifetime of the volume.
  - Amazon EBS currently supports up to 4000 IOPS per Provisioned IOPS volume.
  - You can stripe multiple volumes together to deliver thousands of IOPS per EC2 instance.

EBS Volumes

- To enable your EC2 instances to fully utilize the IOPS provisioned on an EBS volume:
  - Launch selected Amazon EC2 instance types as “EBS-Optimized” instances.
  - EBS-optimized instances deliver dedicated throughput between Amazon EC2 and Amazon EBS, with options between 500 Mbps and 1000 Mbps depending on the instance type used.
- EBS charges are based on per GB-month AND per 1 million I/O requests

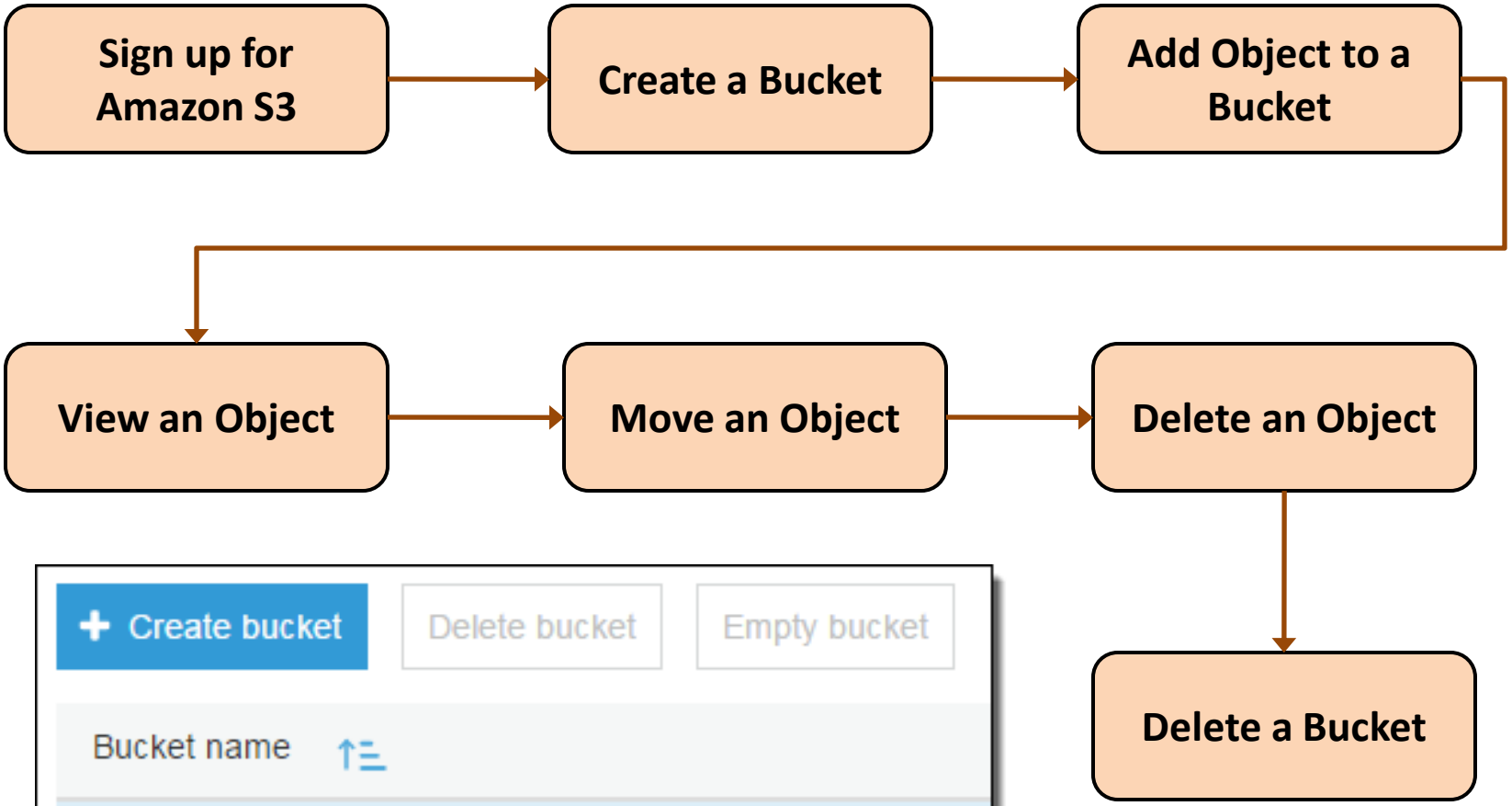
Amazon Simple Storage Service (S3)

- Amazon S3 is a web service to store and retrieve any amount of data, at any time, from anywhere on the web.
- Write, read, and delete objects containing from 1 byte to 5 terabytes of data each.
- However, up to 5GB can be stored per PUT request.
- The number of objects you can store is unlimited.
- Authentication mechanisms are provided to ensure that data is kept secure from unauthorized access.
- Objects can be made private or public, and specific access rights can be granted to specific users.
- S3 charges are based on per GB-month AND per I/O requests AND per data modification requests.

Amazon Simple Storage Service (S3)

- Each object is stored in a bucket and retrieved via a unique, developer-assigned key.
  - Buckets are simply containers for objects, and provide similar access as Internet names: Ex: mybkt.s3.amazonaws.com
  - A KEY is a unique identifier for an object stored in a bucket
  - Bucket and KEY together uniquely identify each object in S3
  - A key can be maximum 1024 bytes long (utf-8) and can contain only allowed characters (Refer documentation)
  - The / (slash) character used in key name automatically creates folders in the S3 bucket.
- A bucket can be stored in one of several Regions.
- Customer can choose a Region to optimize for latency, minimize costs, or address regulatory requirements.

Amazon Simple Storage Service (S3)



Amazon Simple Storage Service (S3)

- Advantages:
  - Scalability: amount of storage & bandwidth can scale without any changes to configuration.
  - Availability, speed, throughput, capacity, and robustness is not affected even if you gain 10,000 users overnight.
  - Unlimited storage, and you pay as you go.
  - Inexpensive, no capital expenditure needed.
  - Data is accessible from any location on Internet.
  - More reliable than any other cheap storage providers.
- Disadvantages
  - Trust: one may not feel comfortable to store sensitive and confidential data on the cloud. e.g., banking transactions.
  - Any outage of S3 may affect a critical business.

Message Queues (Amazon SQS)

- A message queue is an asynchronous service-to-service communication used in serverless architectures.
- Applications are decoupled into smaller, independent components → easier to develop, deploy and maintain.
  - Message queues provide communication and coordination for these distributed applications.
  - Message queues allow different parts of a system to communicate and process operations asynchronously.
  - Messages are usually small, and can be things like requests, replies, error messages, or just plain information.
  - Messages are stored in the queue until they are processed.
  - Each message is processed only once, by a single consumer.
  - Once a message is processed, it is deleted from the queue.

Amazon Simple Queue Service (SQS)

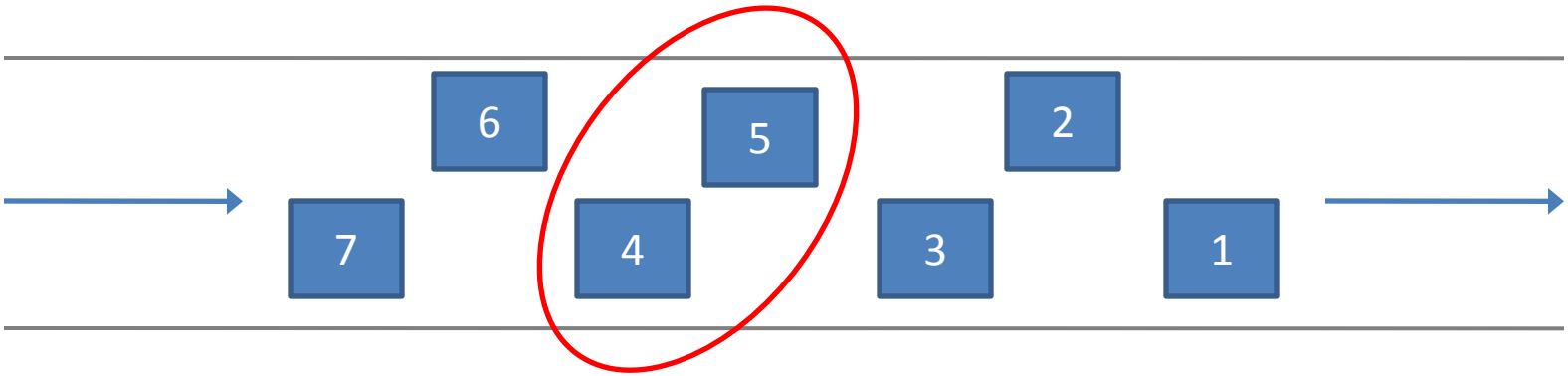
- To send a message, a component called a producer adds a message to the queue.
- The message is stored on the queue until another component called a consumer retrieves the message and processes it.
- Many producers and consumers can use the queue, but each message is processed only once, by a single consumer → 1-to-1 or point-to-point communication.

Amazon Simple Queue Service (SQS)

- Amazon SQS is a secure, durable, and available queue service to integrate and decouple distributed software systems and components.
- It provides a generic web services API that can be used in any programming language supported by AWS SDK.
- Amazon SQS supports both Standard and FIFO queues.
- Standard queue is available in all regions.
- FIFO queues are available only in the following Regions:
  - US East (N. Virginia), US East (Ohio), US West (Oregon)
  - EU (Ireland) Regions

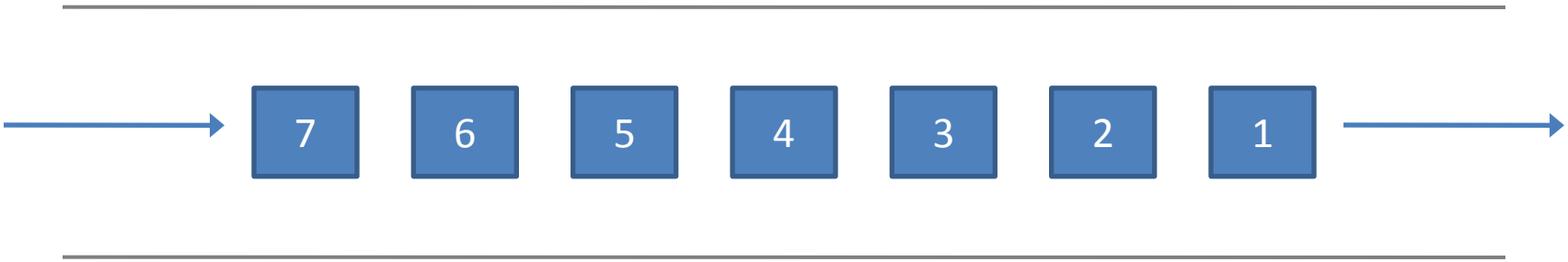
SQS – Features of Standard Queue

- **Unlimited Throughput** – Standard queues support a nearly unlimited TPS (Txn/sec) per action.
- **At-Least-Once Delivery** – A message is delivered at least once, but occasionally more than one copy of a message may be delivered.
- **Best-Effort Ordering** – Occasionally, messages might be delivered in a different order in which they were sent.



SQS – Features of FIFO Queue

- **High Throughput** – By default, FIFO queues support up to 3,000 messages per second with batching.
  - FIFO queues support up to 300 messages per second (300 send, receive, or delete operations per second) without batching.
- **Exactly-Once Processing** – A message is delivered once and remains until a consumer processes it (no duplicates)
- **First-In-First-Out Delivery** – The order in which messages are sent and received is strictly preserved.



Benefits of Amazon SQS

- **Security** – Who can send messages and who can receive messages from Amazon SQS queue is strictly controlled.
  - Server-side encryption (SSE) allows transmission of sensitive data by protecting the contents of messages.
- **Durability** –Amazon SQS stores messages on multiple servers.
  - Standard queues support at-least-once message delivery.
  - FIFO queues support exactly-once message processing.
- **Availability** – Amazon SQS uses redundant infrastructure to provide highly-concurrent access to messages and high availability for producing and consuming messages.

Benefits of Amazon SQS

- **Scalability** – SQS can process each buffered request independently, scaling transparently to handle any load increases or spikes without any provisioning instructions.
- **Reliability** – SQS locks the messages during processing
  - Multiple producers can send and multiple consumers can receive messages at the same time.
- **Customization** – Queues don't need to be exactly alike
  - Customer can set a default delay on a queue.
  - Contents of messages > 256 KB can be stored on S3
  - SQS will only hold a pointer to the Amazon S3 object
  - A large message can also be split into smaller messages.

VMware vCloud

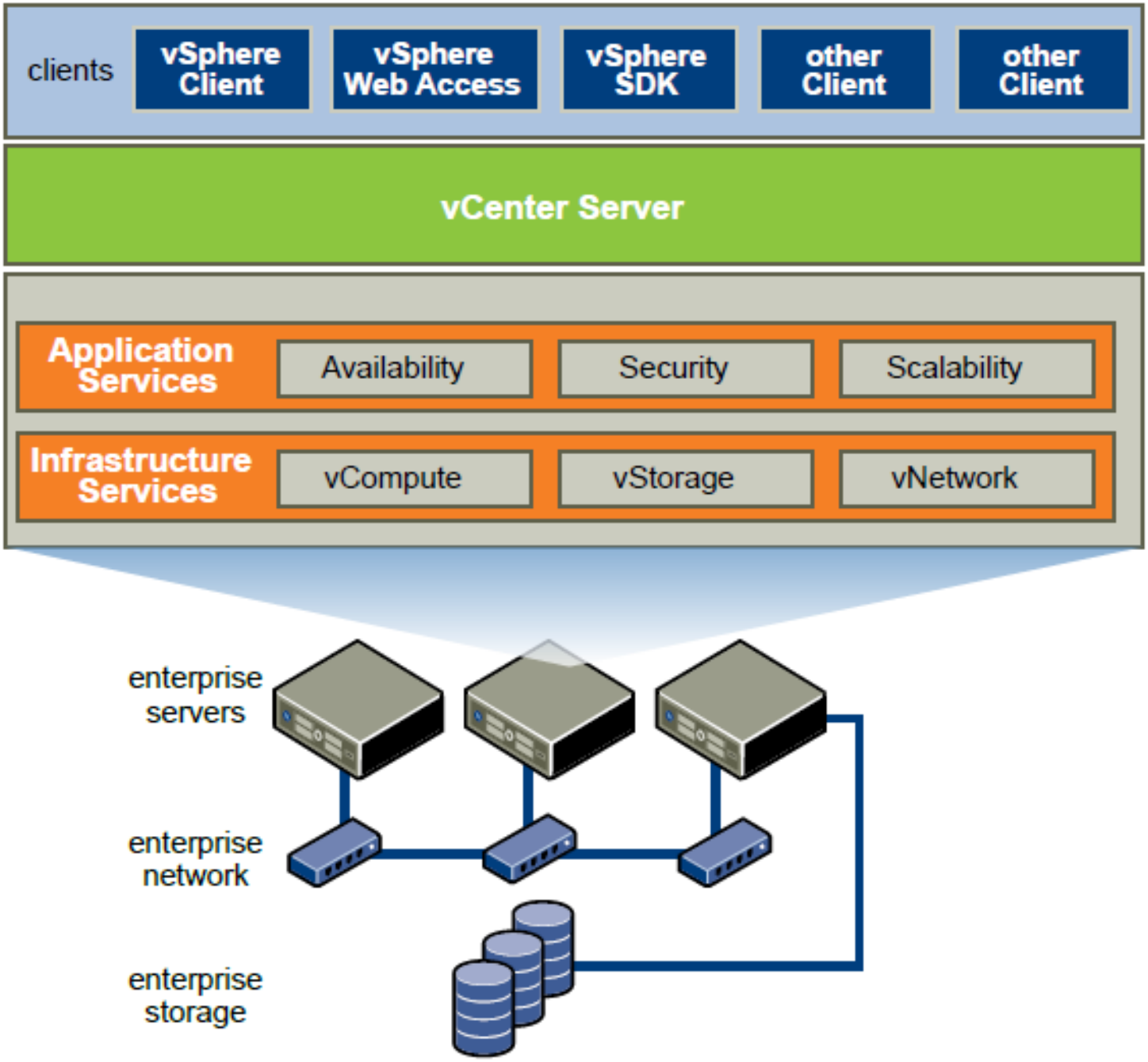
- VMware vCloud *Suite* is an enterprise-ready, cloud management platform that includes:
  - **vSphere**: an industry leading server virtualization platform.
  - **vRealize**: a cloud management platform.
    - ***vRealize Operations***: Intelligent performance, capacity, and configuration management for multi-vendor environments.
    - ***vRealize Automation***: Self-service, policy-based infrastructure and application provisioning and lifecycle management for public, private, or hybrid cloud environments
    - ***vRealize Business***: Automated costing, usage metering, and service pricing of multi-vendor virtualized environments.
- vSphere leverages the power of virtualization to transform datacenters into simplified cloud computing infrastructures.

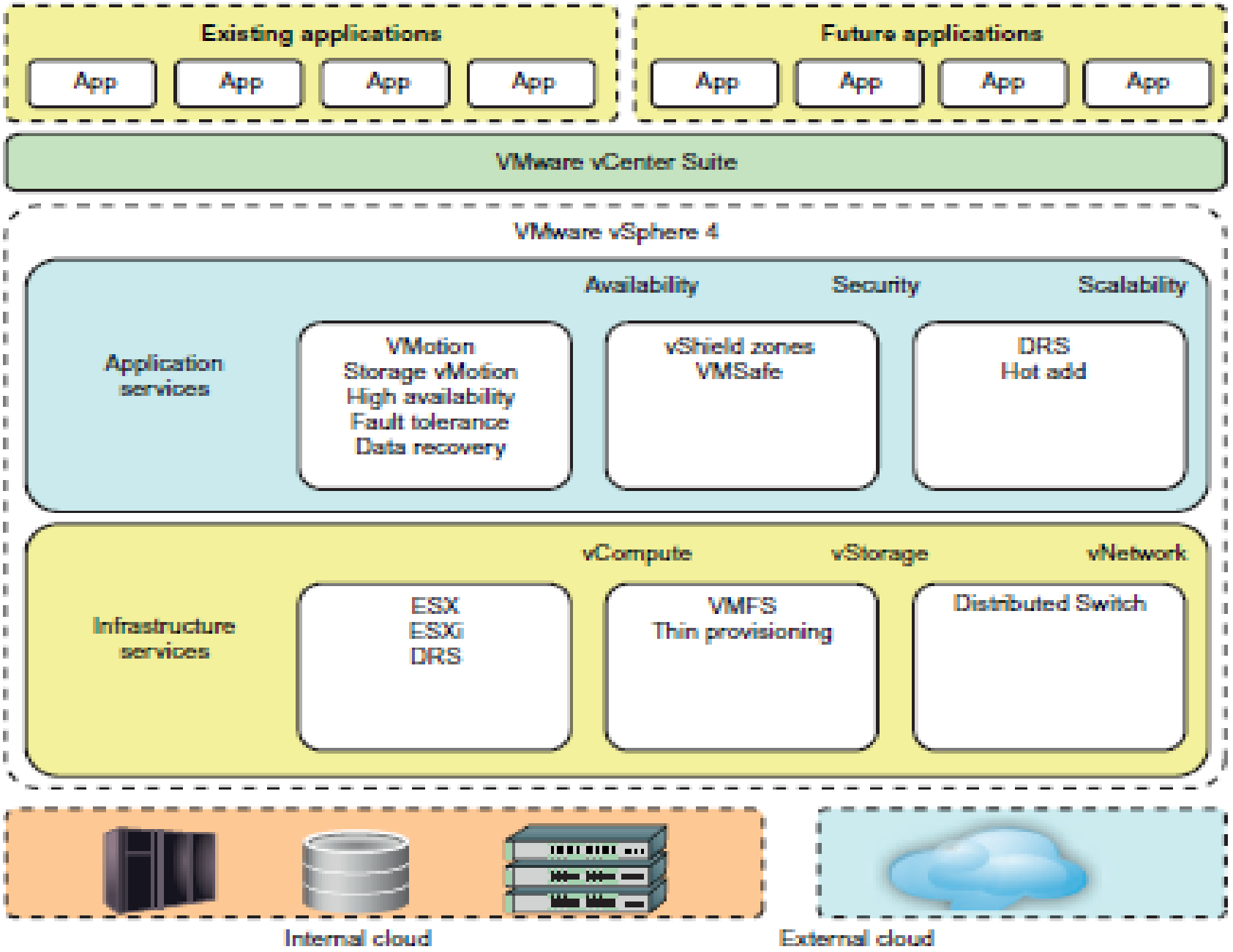
Components of VMware vSphere

- Infrastructure Services:
  - set of services provided to abstract, aggregate, and allocate hardware or infrastructure resources
    - **vCompute**: enables aggregation of disparate server resources across many discrete servers and assign them to applications.
    - **vStorage**: provides a set of technologies that enables the most efficient use and management of storage in virtual environments
    - **vNetwork**: provides a set of technologies that simplify and enhance networking in virtual environments
- Application Services:
  - It is a set of services provided to ensure availability, security, and scalability for applications
    - e.g., High Availability and Fault Tolerance

Components of VMWare vSphere

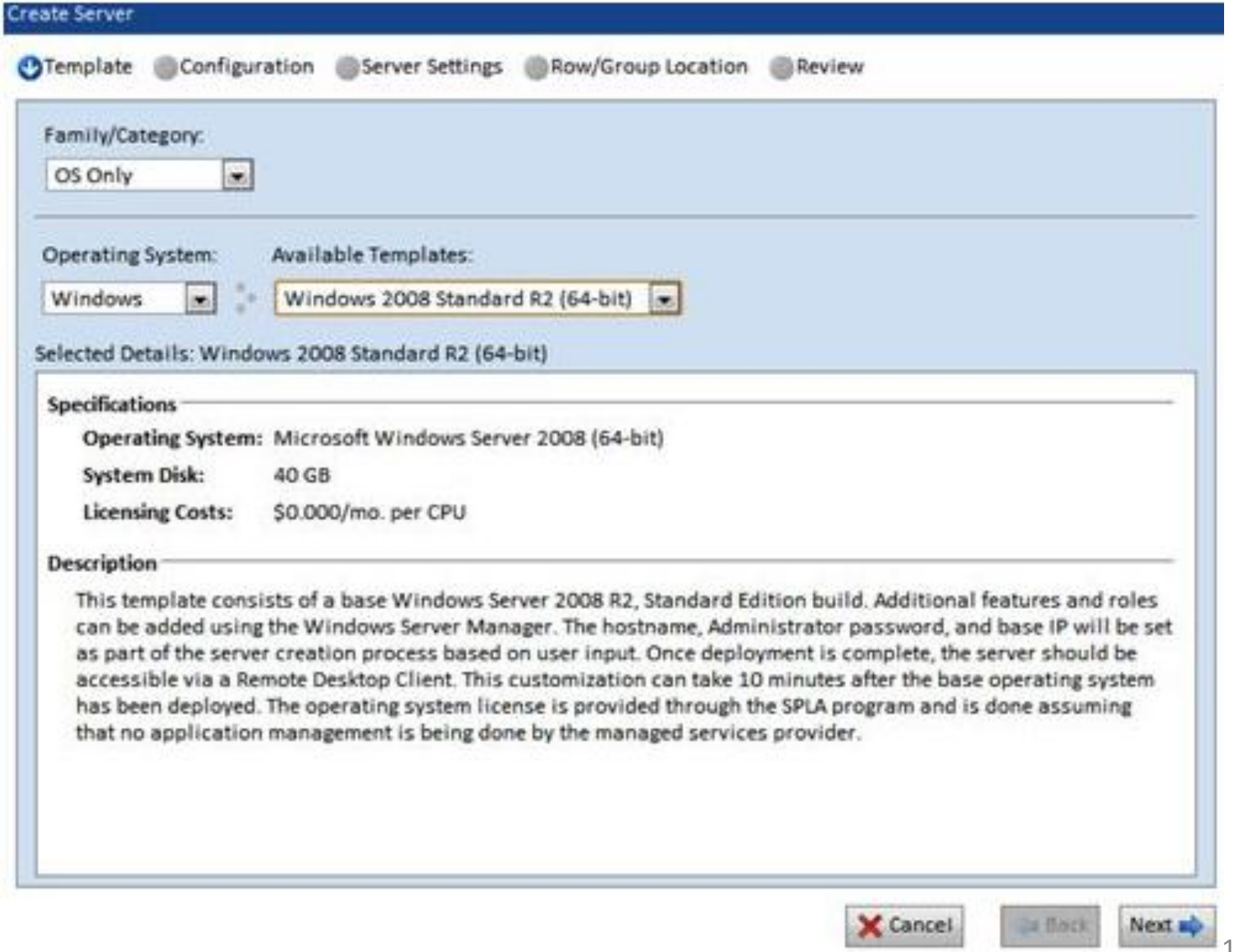
- vCenter Server:
  - provides a single point of control of the essential datacenter services such as access control, performance monitoring, and configuration.
- Clients:
  - Users can access the VMware vSphere datacenter through clients such as the vSphere Client or Web Access through a web browser.
- Hypervisors
  - VMware ESXi – Runs directly on hardware (bare-metal, Type-I)
  - VMware GSX – Requires a host operating system (Type-II)
- VMFS (Virtual Machine File System)
  - High performance cluster file system for ESX/ESXi virtual machines
- VMotion
  - VMotion enables the live migration of running virtual machines from one physical server to another with zero down time

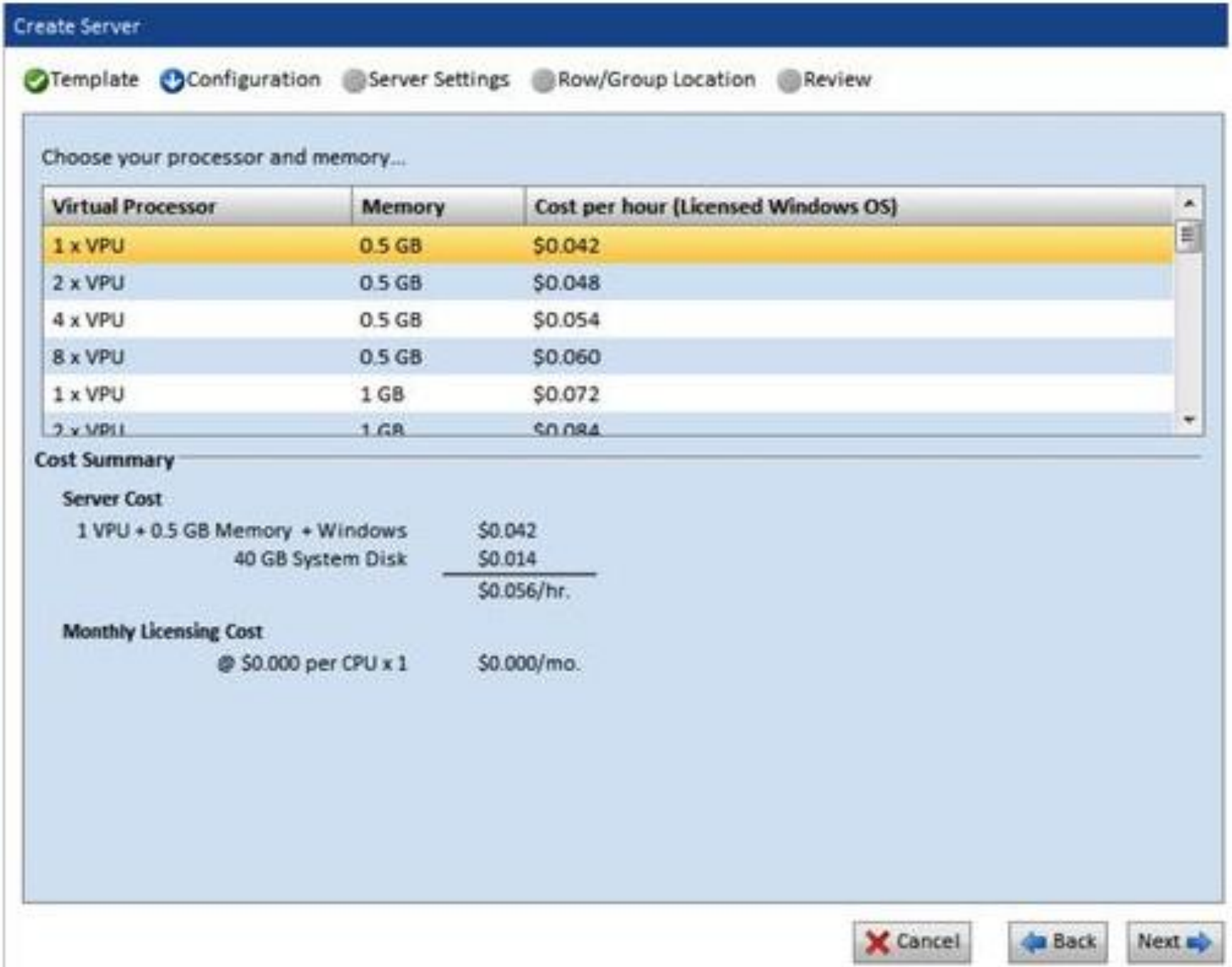


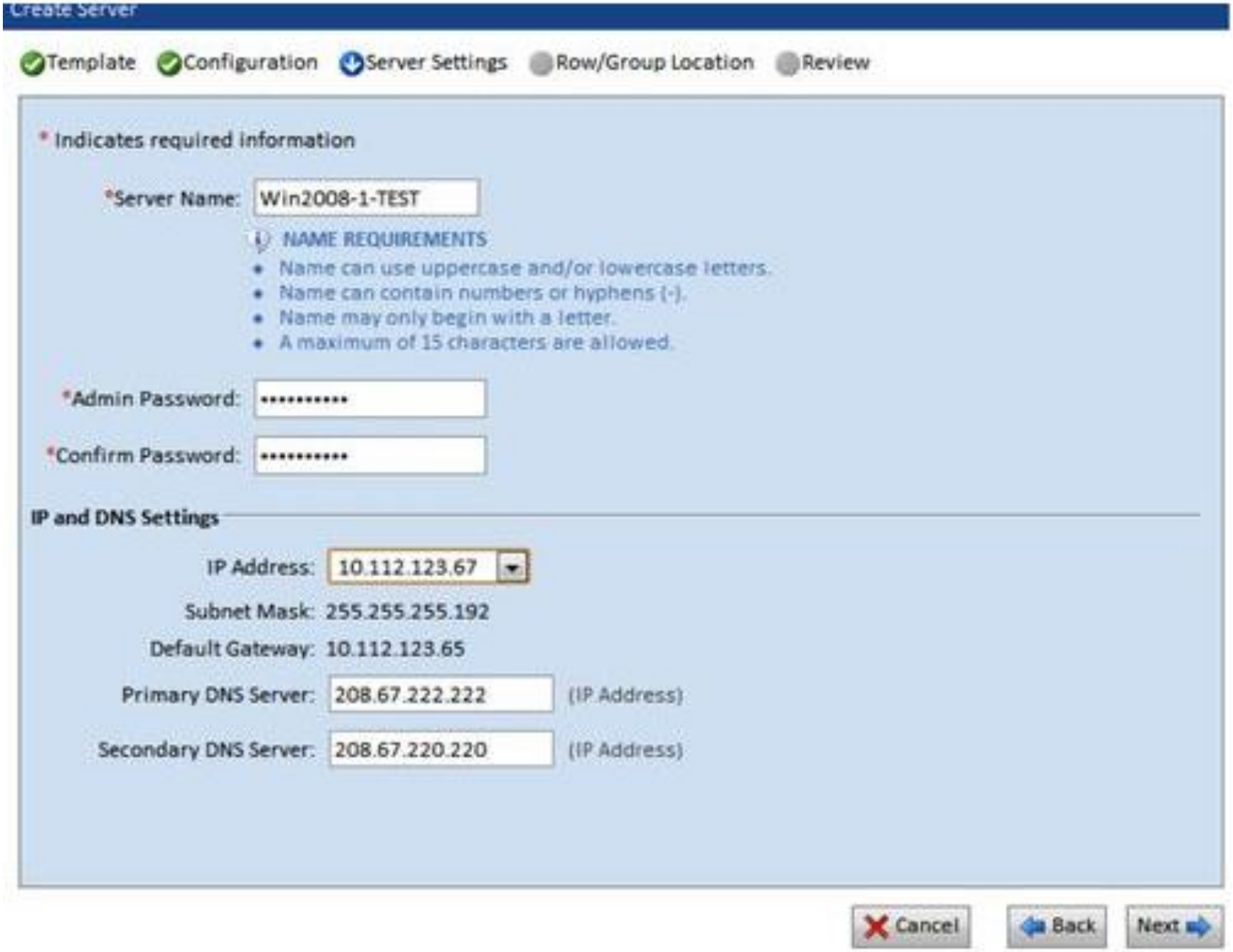


vCloud Express

- It is VMware's partner-provided Cloud Infrastructure services (IaaS) using VMware's product range
  - **Terremark**, Hosting.com, Logica, Melbourne IT, Bluelock etc.
  - No-commitment & pay as you go with a credit card
  - Supports more than 450 host operating systems
  - Up to 8-way 16GB Virtual Machines
  - Supports Windows 2008 and SQL 2008
  - Offers hardware load balancing
  - Fiber-attached persistent storage
  - Easy to set up and use
  - Pricing starts from $0.036 per computing-hour



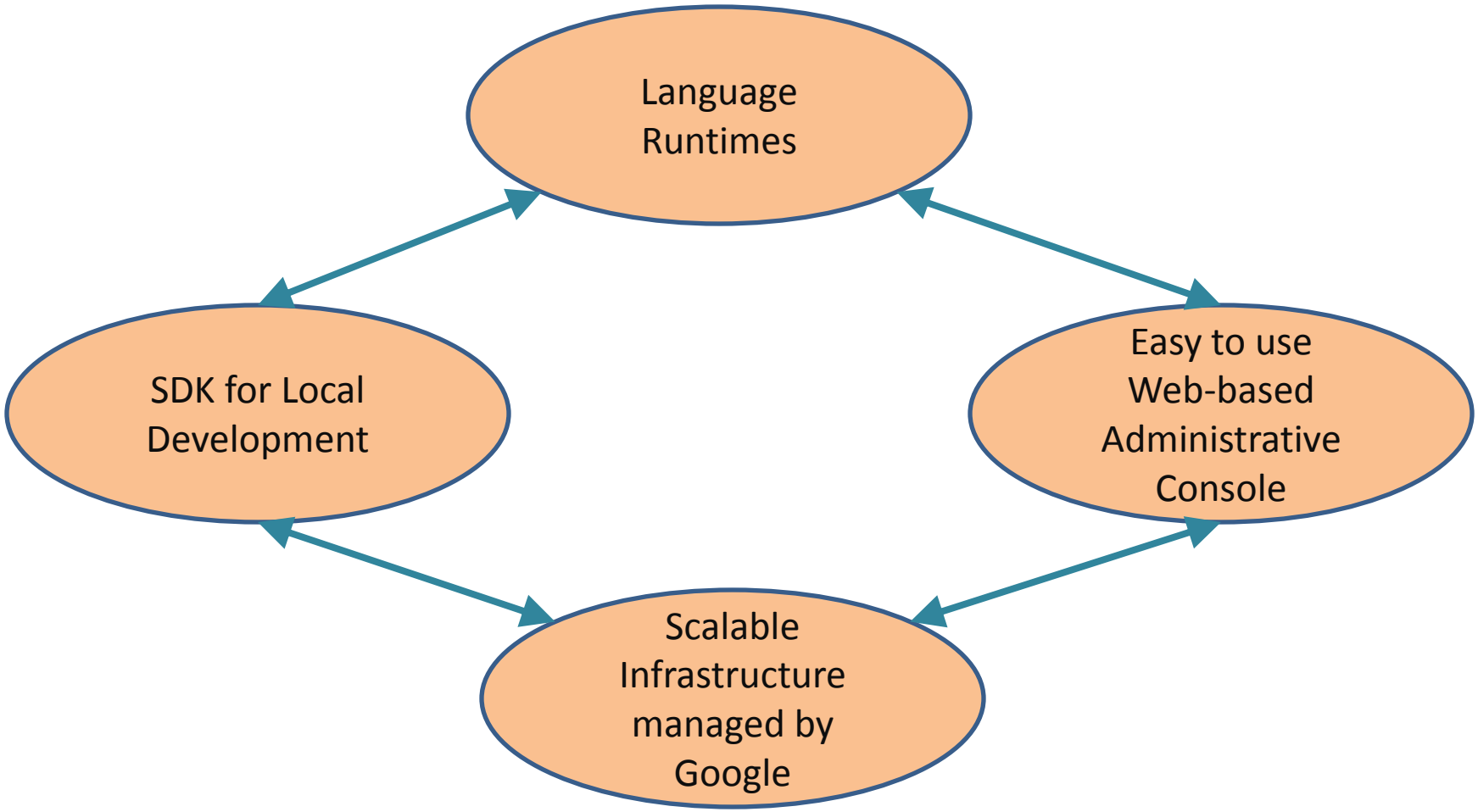




PaaS : Google App Engine (GAE)

- What is it?
  - Google's platform to build web applications on the cloud
  - Dynamic web server with complete support for common web technologies, e.g., Java, Python, PHP …
  - Automatic scaling and load balancing
  - Transactional Data Store model
- Why Google App Engine?
  - Lower TCO (Total Cost of Ownership)
  - Rich set of APIs and Libraries
  - Fully featured SDK for local development
  - Ease of Deployment
  - Managing & monitoring through GAE Admin Console

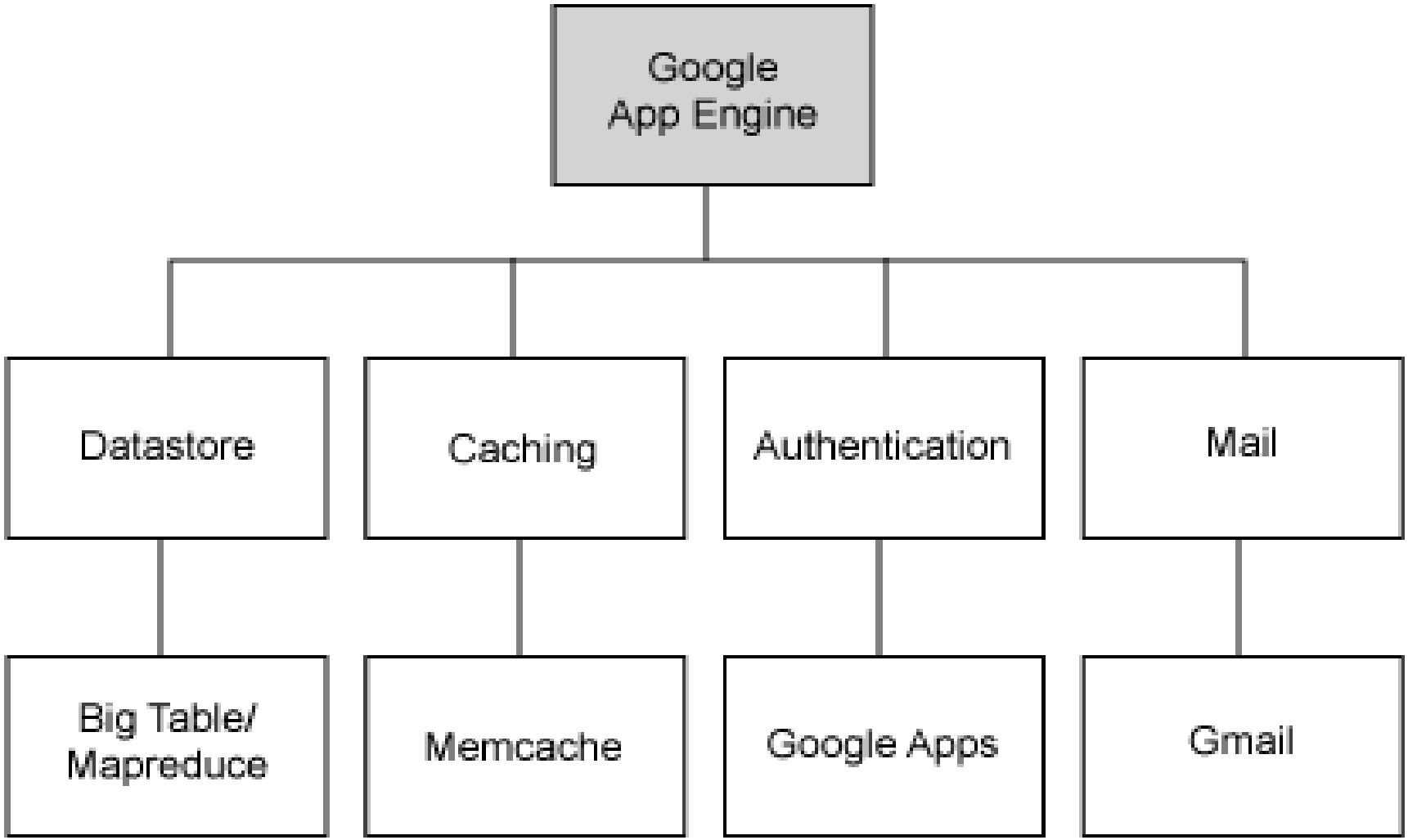
Google App Engine (GAE)



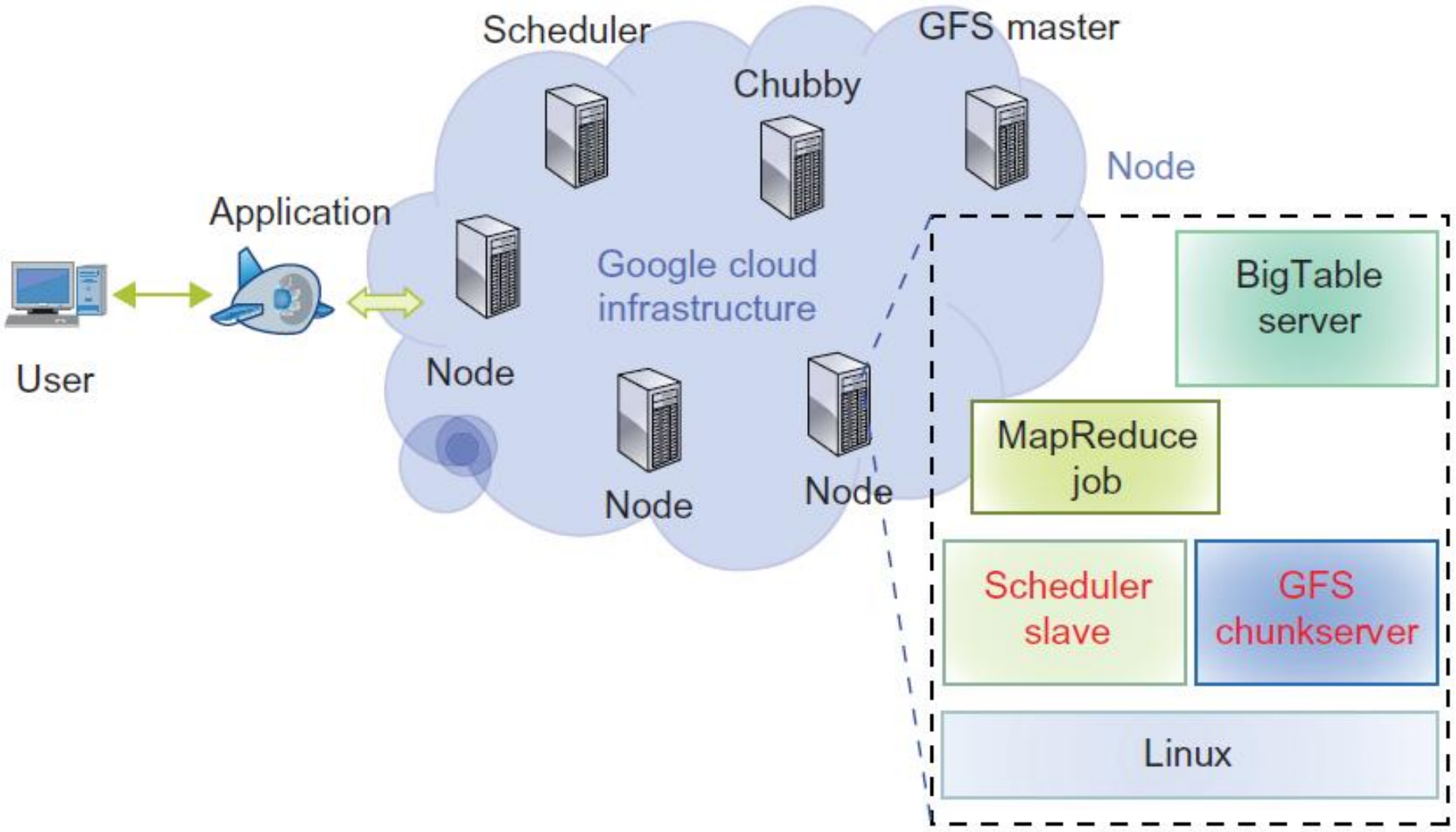
Google App Engine (GAE)

- GAE is a PaaS and cloud computing platform for developing and hosting web applications.
- Applications are hosted in Google-managed data centers.
- Applications are sandboxed and run on multiple servers.
- GAE offers automatic scaling for web applications – more resources are allocated automatically to meet demand.
- GAE is free up to:
  - 1GB of storage, enough CPU & Bandwidth for supporting up to 5 million page views per month
  - 25 web applications free per user account
- Provides web application development and hosting platform using Java, Python, PHP, Node.js, Go, and more.

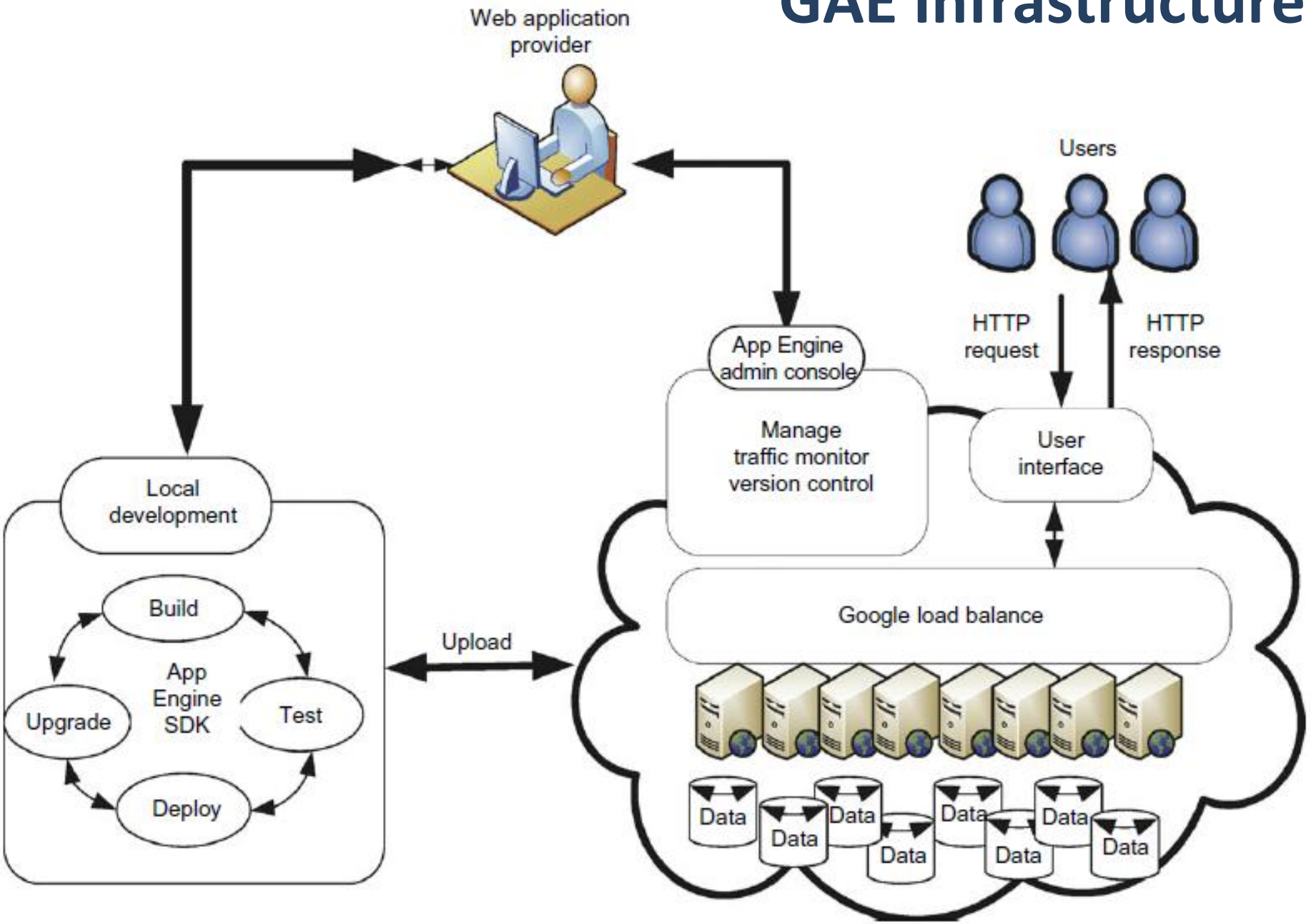
Google App Engine (GAE)



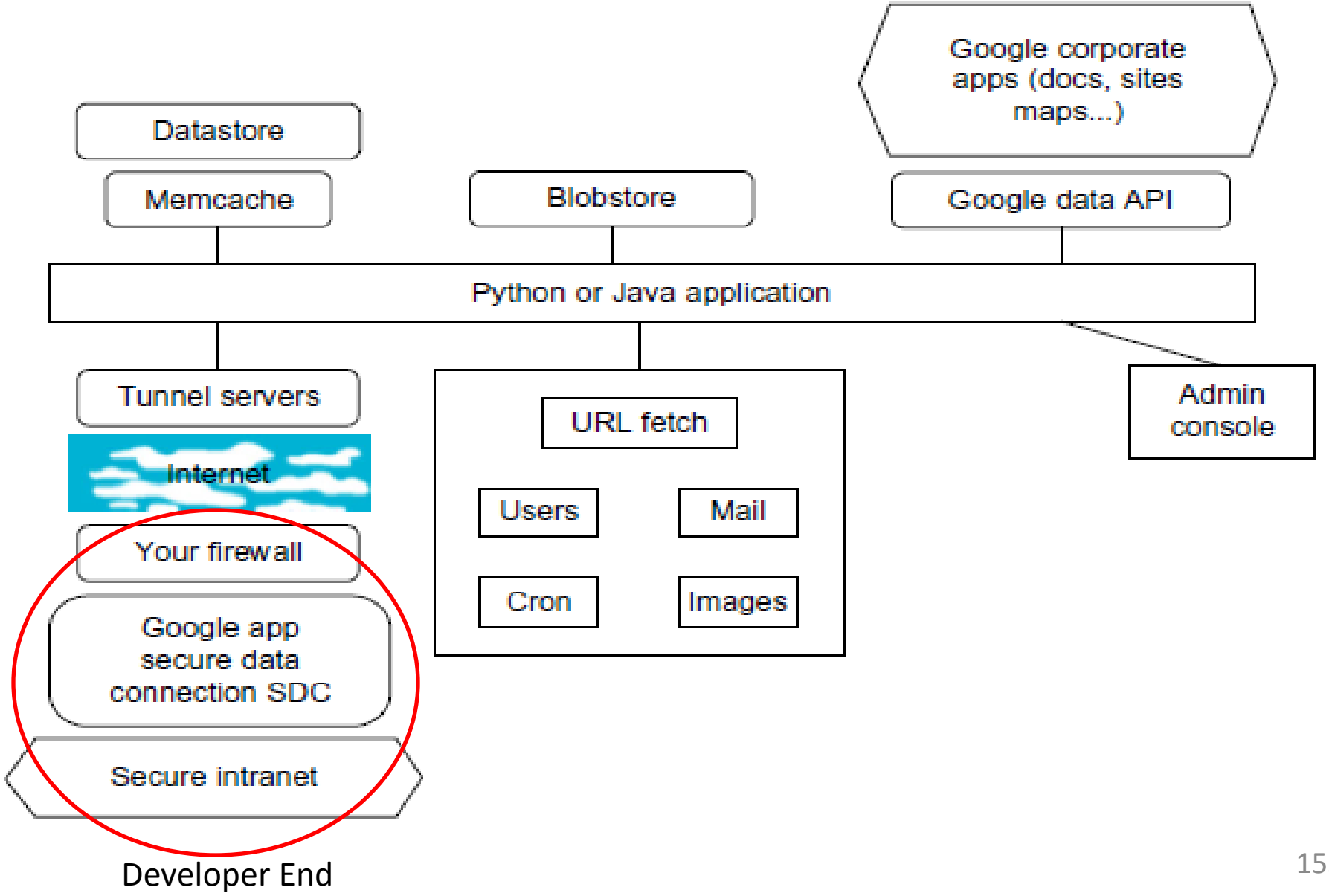
GAE Infrastructure

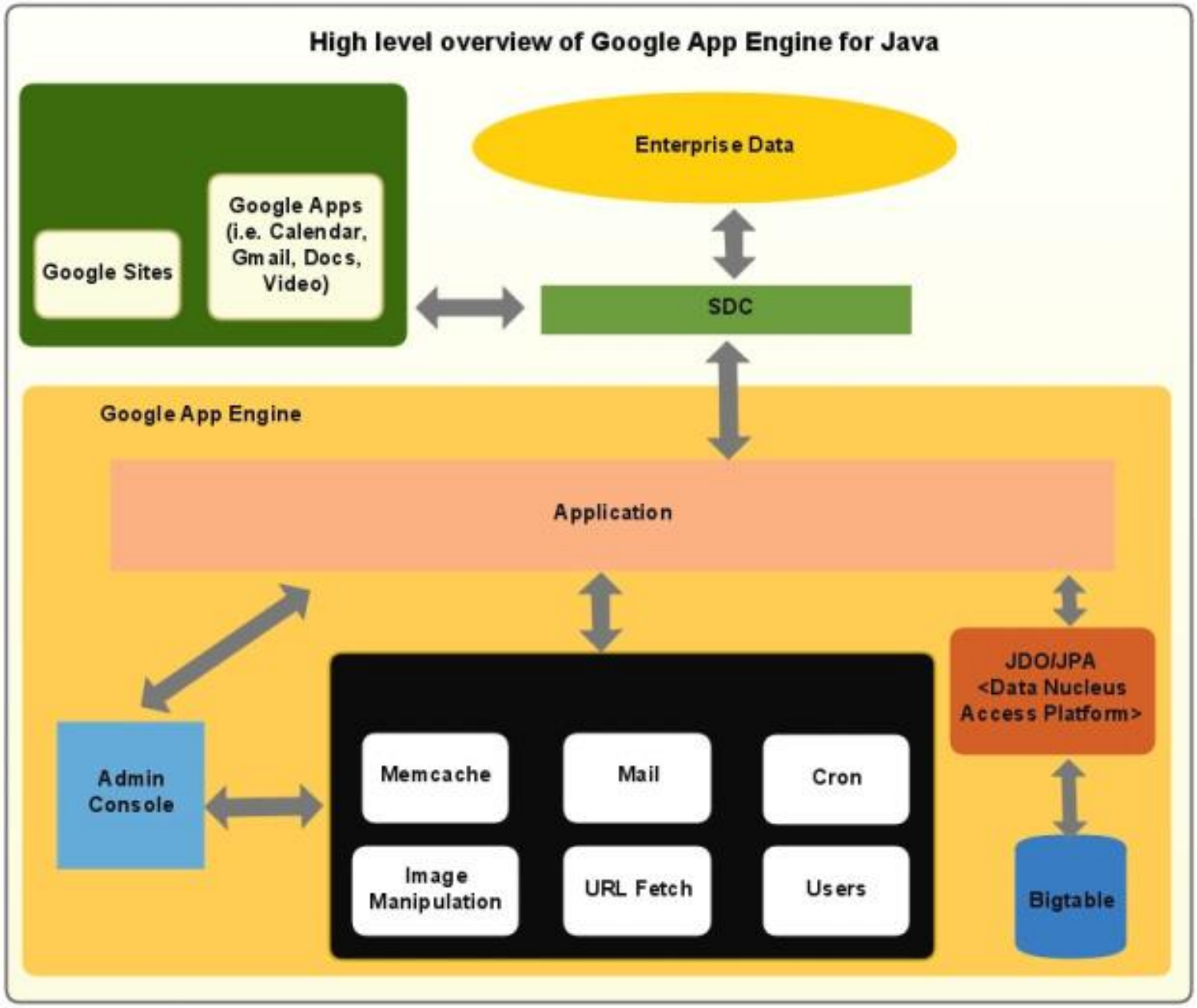


GAE Infrastructure

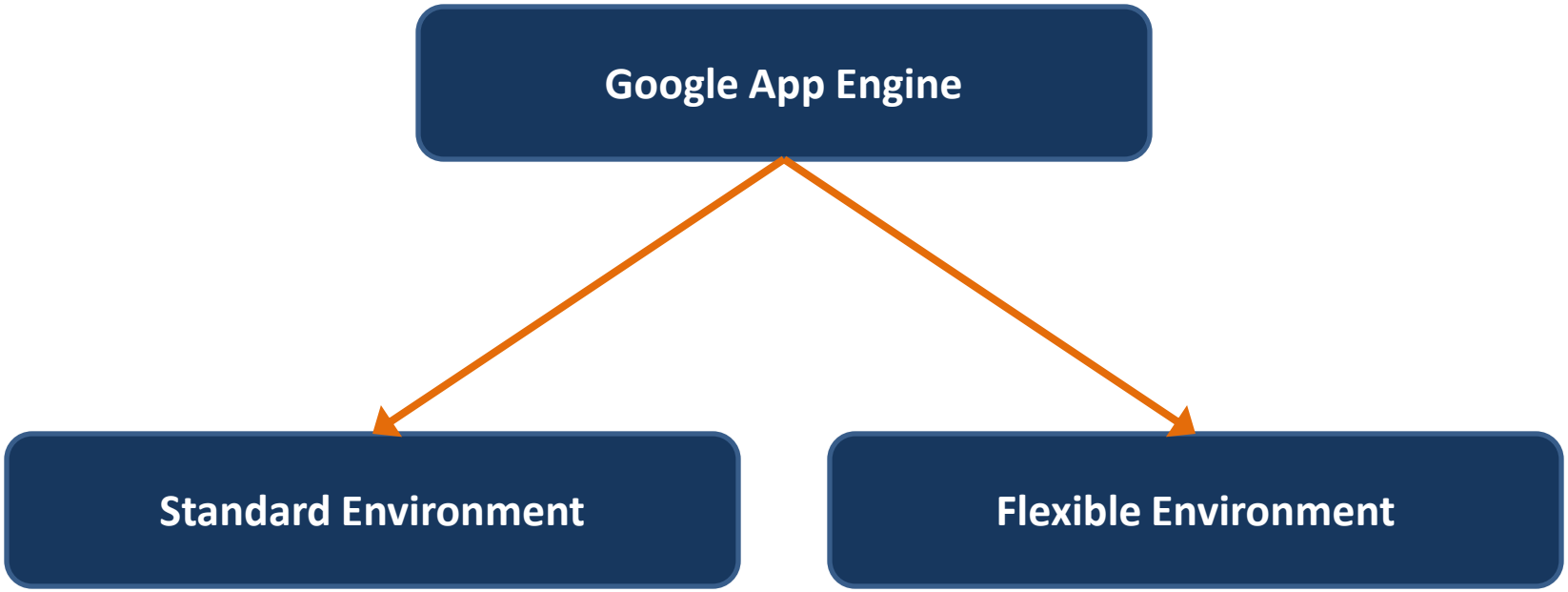


GAE Programming Environment





GAE Environments



Google App Engine (GAE)

- Two types of environments are available for GAE
  - Standard Environment
    - It is based on container instances running on Google's infrastructure.
    - Containers are preconfigured with one of several available runtimes (Java 7, Java 8, Python 2.7, Python 3.6, Go, PHP and Node.js)
    - Customization of runtime or OS is not allowed.
  - Flexible Environment
    - VM instances are based on Google Compute Engine
    - Automatically scales your app up and down while balancing the load.
    - Microservices, authorization, SQL and NoSQL databases, traffic splitting, logging, versioning, security scanning, and content delivery networks are all supported natively.
    - Allows you to customize the runtime and even the operating system of your virtual machine.
    - Includes native support for Java 8, Python 2.7 and Python 3.6, Node.js, Ruby, PHP, .NET Core, and Go.

Google App Engine

- Google manages the virtual machines, ensuring that:
  - Instances are health-checked, healed as necessary, and co-located with other services within the project.
  - Critical, backward-compatibility updates are automatically applied to the underlying operating system.
  - VM instances are automatically located by geographical region according to the settings in your project.
  - Google's management services ensure that all of a project's VM instances are co-located for optimal performance.
  - VM instances are restarted on a weekly basis. During restarts Google's management services will apply any necessary operating system and security updates.
  - Root access is provided to Compute Engine VM instances.
  - SSH access to VM instances in the flexible environment is disabled by default.

Instances in GAE Standard Environment

Instance Class	Memory Limit	CPU Limit	Supported Scaling Types
F1 (default)	128 MB	600 MHz	automatic
F2	256 MB	1.2 GHz	automatic
F4	512 MB	2.4 GHz	automatic
F4_1G	1024 MB	2.4 GHz	automatic
B1 (default)	128 MB	600 MHz	manual, basic
B2	256 MB	1.2 GHz	manual, basic
B4	512 MB	2.4 GHz	manual, basic
B4_1G	1024 MB	2.4 GHz	manual, basic
B8	1024 MB	4.8 GHz	manual, basic

GAE Pricing

Resource	Unit	Unit cost (in US $)
Instances*	Instance hours	$0.05
Outgoing Network Traffic	Gigabytes	$0.12
Incoming Network Traffic	Gigabytes	Free
Datastore Storage	Gigabytes per month	$0.18
Blobstore, Logs, and Task Queue Stored Data	Gigabytes per month	$0.026
Dedicated Memcache	Gigabytes per hour	$0.06
Logs API	Gigabytes	$0.12
SSL Virtual IPs** (VIPs)	Virtual IP per month	$39.00
Sending Email, Shared Memcache, Pagespeed, Cron, APIs (URLFetch, Task Queues, Image, Sockets, Files, and Users)		No Additional Charge

Note: Pricing is applicable only beyond free limits

GAE – Java Runtime Environment

- App Engine runs Java web application using a Java 7 & 8 JVM in a safe sandboxed environment. (Java 7 runtime is being deprecated).
- App Engine invokes the app's servlet classes to handle requests and prepare responses in this environment.
- The secured sandbox environment isolates the application from others for service and security.
- The App Engine platform provides many built-in API services that can be called from your java code.
- The application can also configure scheduled tasks that run at specified intervals.

GAE – Java Runtime Environment

- The App Engine Java 8 runtime is based on OpenJDK 8 and supports all existing features of Java 7 runtime.
  - Does not impose a security manager like Java 7 runtime
  - Supports the standard public Java library
  - Uses Jetty 9 as the HTTP Server and javax.servlet container
  - Supports the Java Servlet 3.1 and 2.5 specifications
  - Supports all Google Cloud-based APIs accessible from the Google Cloud Client Library for Java
  - GAE Java runtime distributes requests for applications across multiple web servers and prevents one application from interfering with another.
  - Web requests to an application must be handled within 60 secs.

GAE – Python Runtime Environment

- GAE provides Python runtime environments to develop web applications using Python programming language.
- Supports Python 2.7 and 3.7 (beta)
- GAE executes the Python applications using a pre-loaded Python interpreter in a safe sandboxed environment.
- GAE includes a simple web application framework, called *webapp2*, to make it easy to get started.
- For larger applications, third-party frameworks, such as *Django*, also work well with the Google App Engine.
- The Python interpreter can run any Python code, including Python modules included in the application, along with the Python standard library.
- Code must be pure Python. Modules with C code won't run.

Google File System (GFS)

- **GFS** is a proprietary distributed file system developed by Google to provide efficient & reliable access to data using large clusters of commodity hardware.
- **Motivations for GFS**
  - Petabytes of data, millions of users, lots of services and servers with data centers spanning across the world → *Scalability*
  - Failures are normal and anything can fail at any time (hardware, network, power outage…) → *Fault Tolerant*
  - Monitoring and maintaining large scale infrastructure and huge amount of data – difficult task → *Autonomic Computing*
  - Should be built from commodity hardware → *Low Cost*
  - Expect files from 100's of MBs to GBs → *Wide Support*
  - Support large streaming reads and small random reads

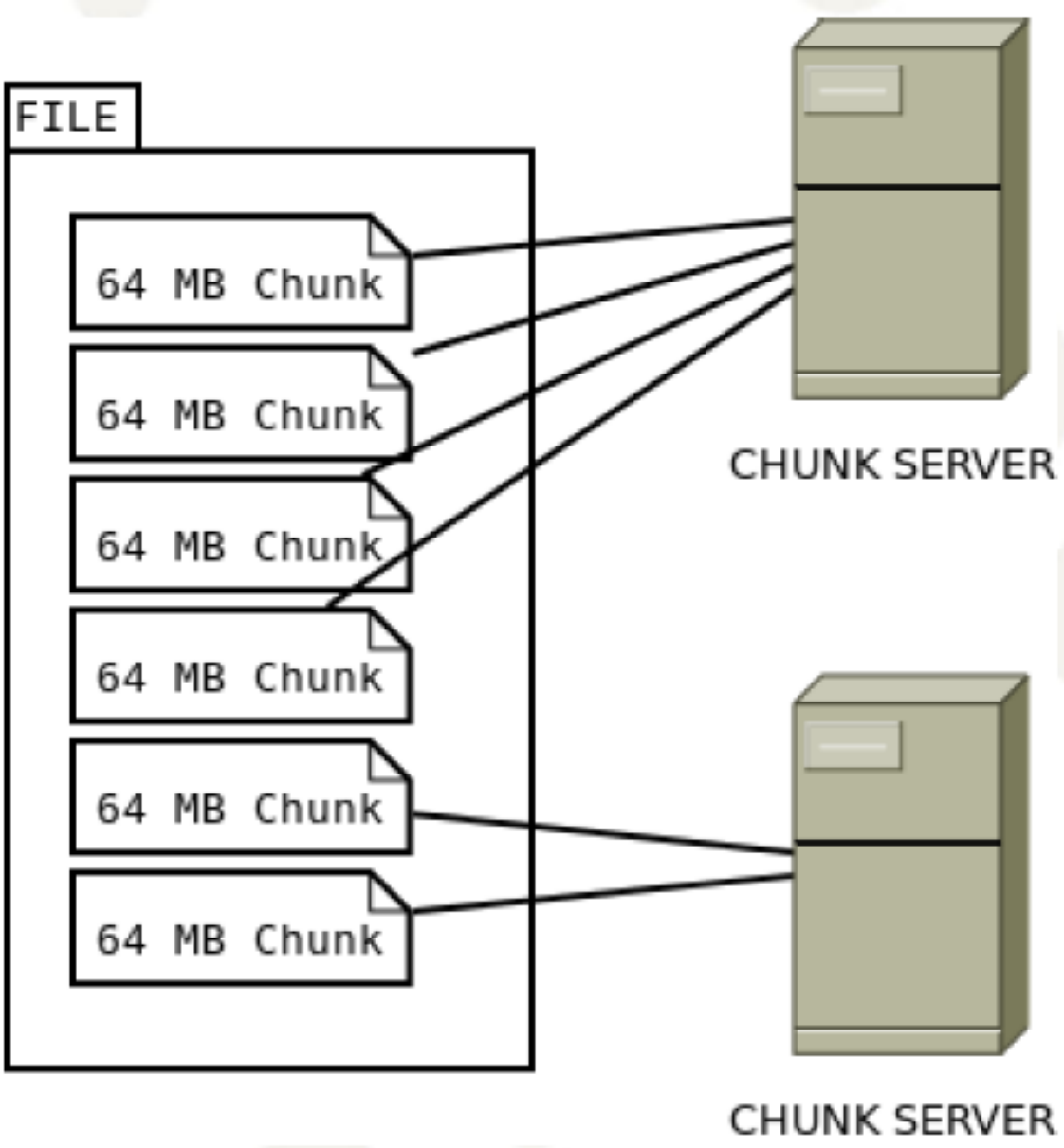
Google File System (GFS)

- **Motivation for GFS** *(Contd..)*
  - Support large sequential file appends
  - Support for producer-consumer queues for many-way merging while maintaining file atomicity
  - Sustain high bandwidth by writing data in bulk
  - Interface should resemble standard file system – hierarchical directories and path names → *Usability*
  - Support all usual file operations: create, delete, move, open, close, read, write, and append → *Usability*
  - Support snapshot copy : to quickly replicate files across servers
  - Support record append – multiple clients should be able to append data to the same file concurrently
- GFS is primarily designed and optimized for system-to-system communication (not for user-to-system)

Google File System (GFS)

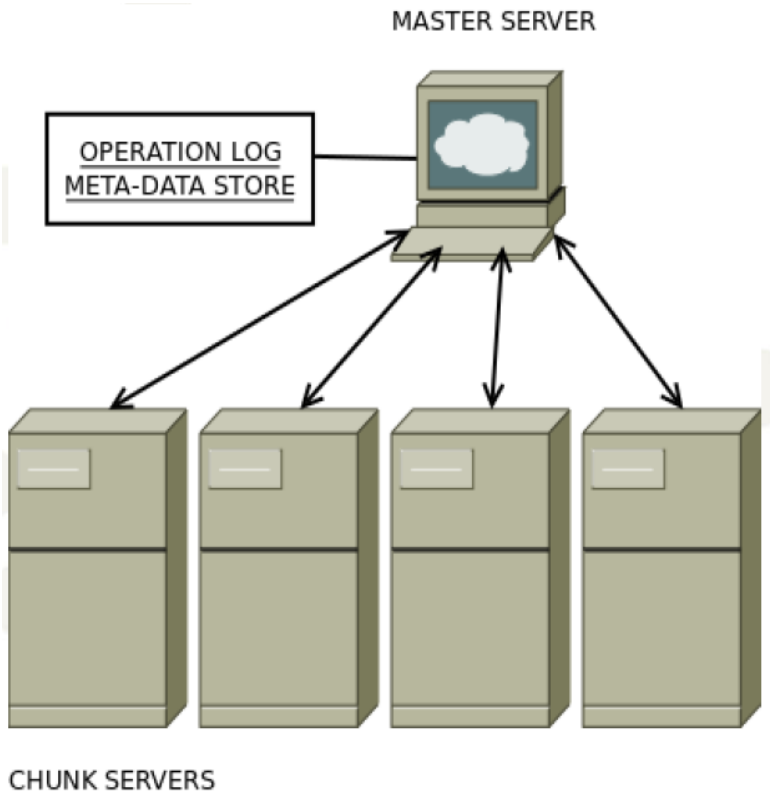
- Files are divided into **chunks**
  - Similar to *blocks* in any other file system, but much larger in size
- Chunk size = 64 MB (fixed)
  - Advantages of large chunk size:
    - Client can perform many operations on a given chunk
    - Reduce interaction between client and master, network overhead
    - Reduce size of metadata stored on the master
    - The metadata can reside in memory for faster access
    - Less fragmentation, easier to manage
- Each chunk has a 64-bit *chunk handle* (ID)
- Chunk servers store and manage the chunks
- A chunk may be replicated on multiple chunk servers
- Master Server manages all the chunk servers

GFS - chunks & chunk servers



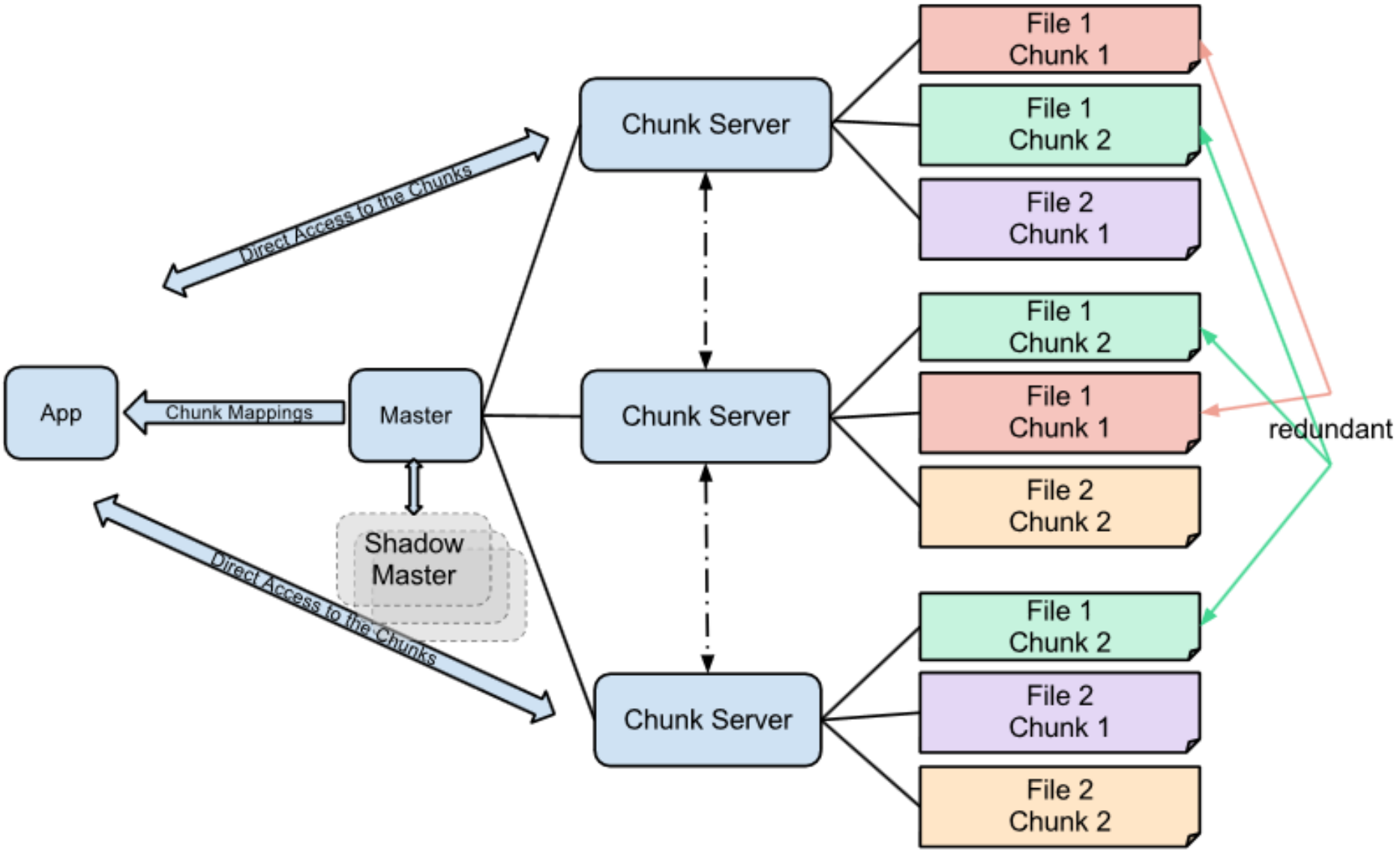
GFS - Master Server

- Coordinates the cluster of ~1000 chunk servers and controls all system-wide activities.
- Maintains file system namespace and locking information.
- Gives "lease" of chunks to one of the chunk servers under its control
- Lease is given to *primary* chunk server, all others holding replicas are called *secondary* chunk servers
- Maintains and updates the operations log – records operations on chunks.
- Stores **meta-data** of the files & chunks and their locations.
- There are redundant shadow master servers to handle failures.



- Master server also handles: chunk lease management, garbage collection of orphaned chunks, and chunk migration.

GFS: A distributed file system



Architecture of GFS

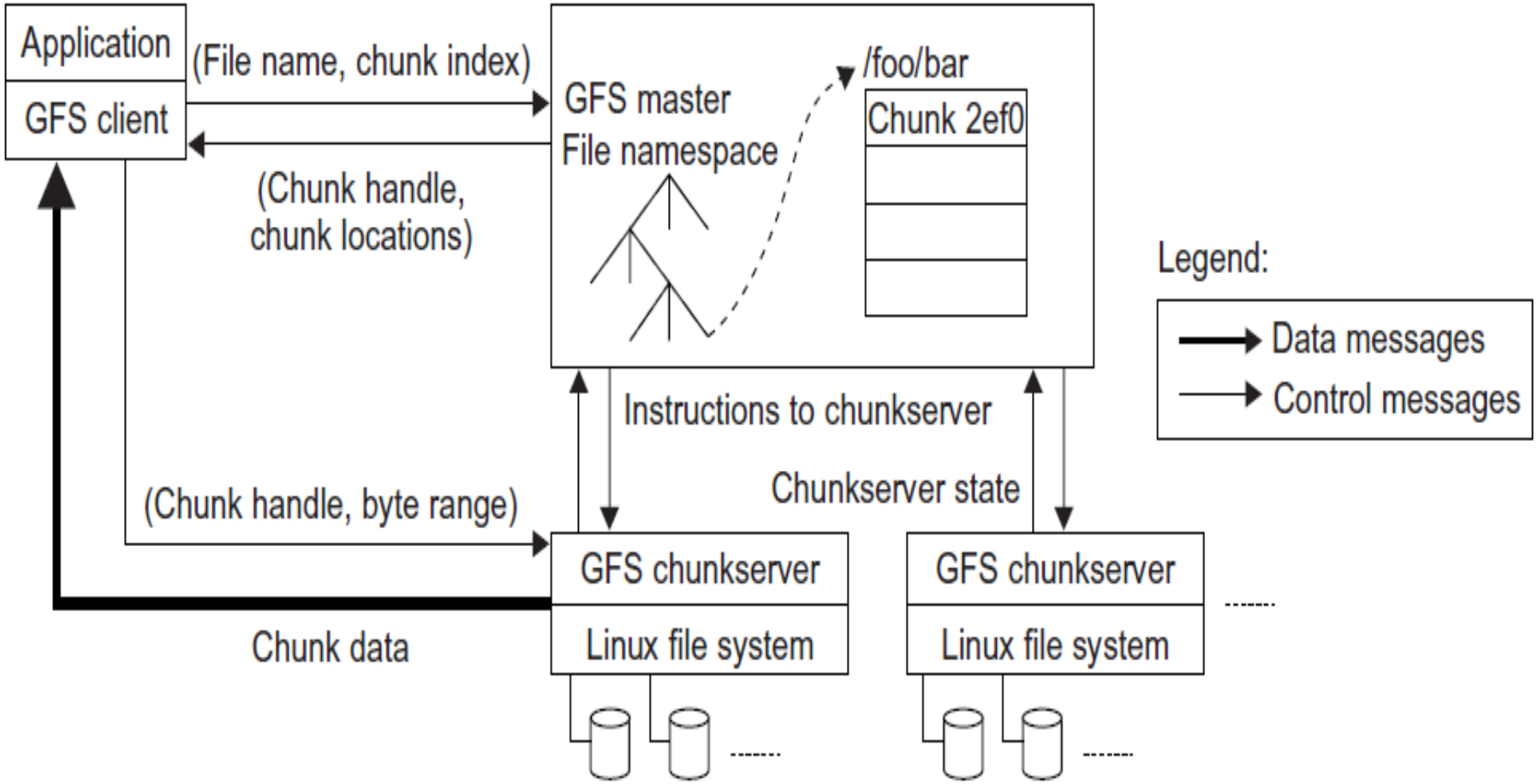


Fig. 6.18 of Hwang, Fox, Dongarra Book

GFS: Metadata

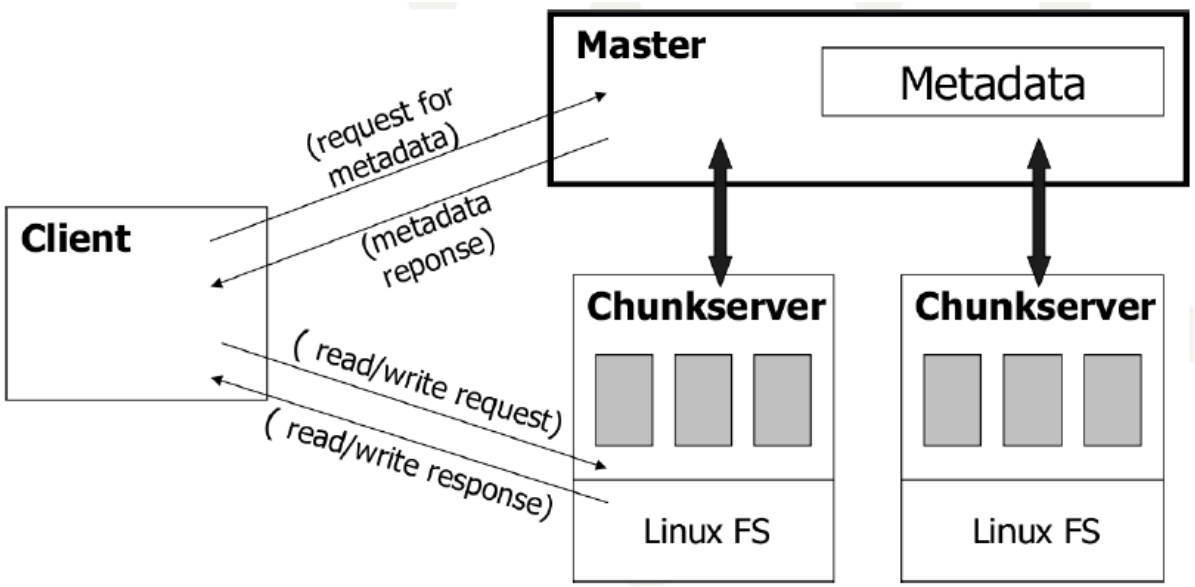
- Master server stores 3 types of major metadata
  - File and Chunk namespace
  - Mapping from file to chunks
  - Location(s) of the replicas of each chunk
- In-memory data structures are used for storing metadata in the Master's memory (for faster access)
- Operations Log: history of operations on a file / chunk
  - Stored reliably on master’s disk
  - Replicated on multiple machines (for durability & consistency)
  - Necessary to re-build system (in case of any catastrophic failure)
  - Check points to speed up recovery (similar to checkpoints found in Log-based recovery mechanism in RDBMS)

GFS: Metadata

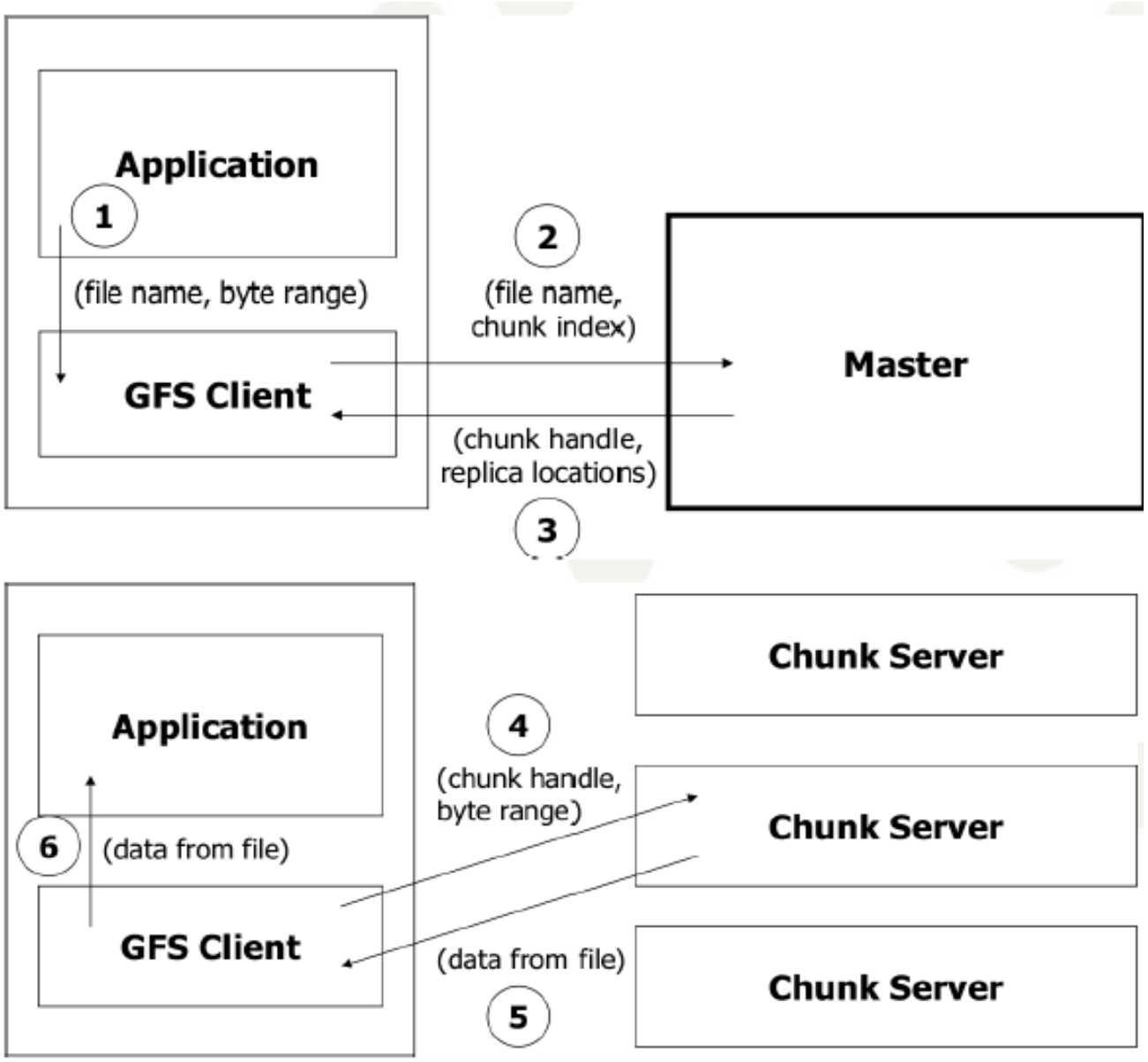
- Chunk Locations:
  - Master does not keep a persistent record of chunk locations
    - Because of chunk server failures, it is not reliable to keep persistent record of chunk locations
  - Instead, it simply polls chunk servers at startup and then periodically (typically every 60 seconds) : *heartbeat* message
  - During heartbeat messages, the state information of the chunk servers is collected by Master server
    - Is the chunk server down?
    - Are there disk failures on the chunk server?
    - Are any chunks (or replicas) corrupted?
    - Which chunk replicas does a chunk server has in its store?
  - Instructions: create new chunk, delete chunk …

GFS : Clients

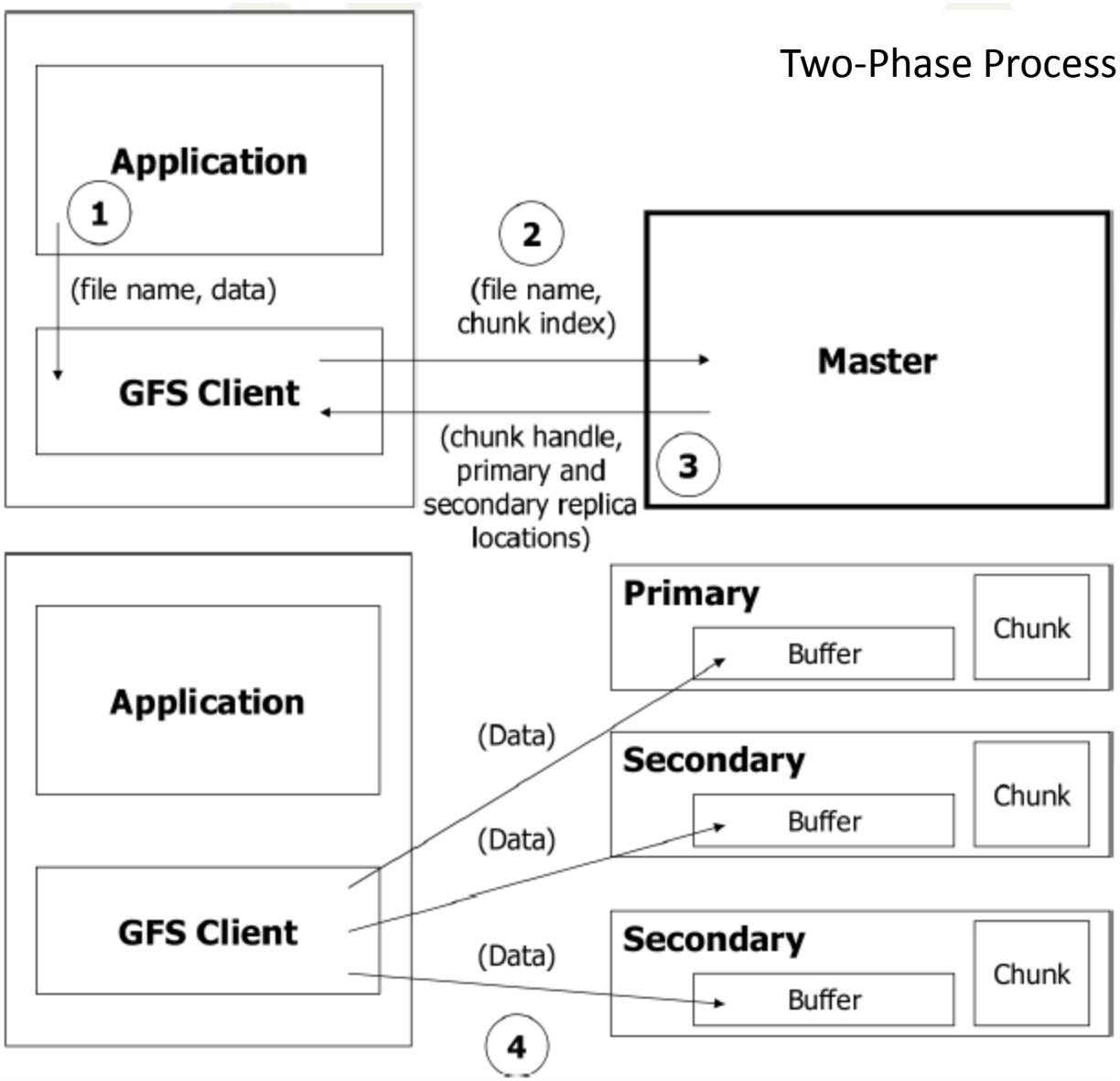
- Clients get only the chunk handle (pointers) from the Master server
- Clients can retrieve data directly from chunk servers following the pointers received from Master
- Clients can cache the pointer information to speed up future accesses → efficiency



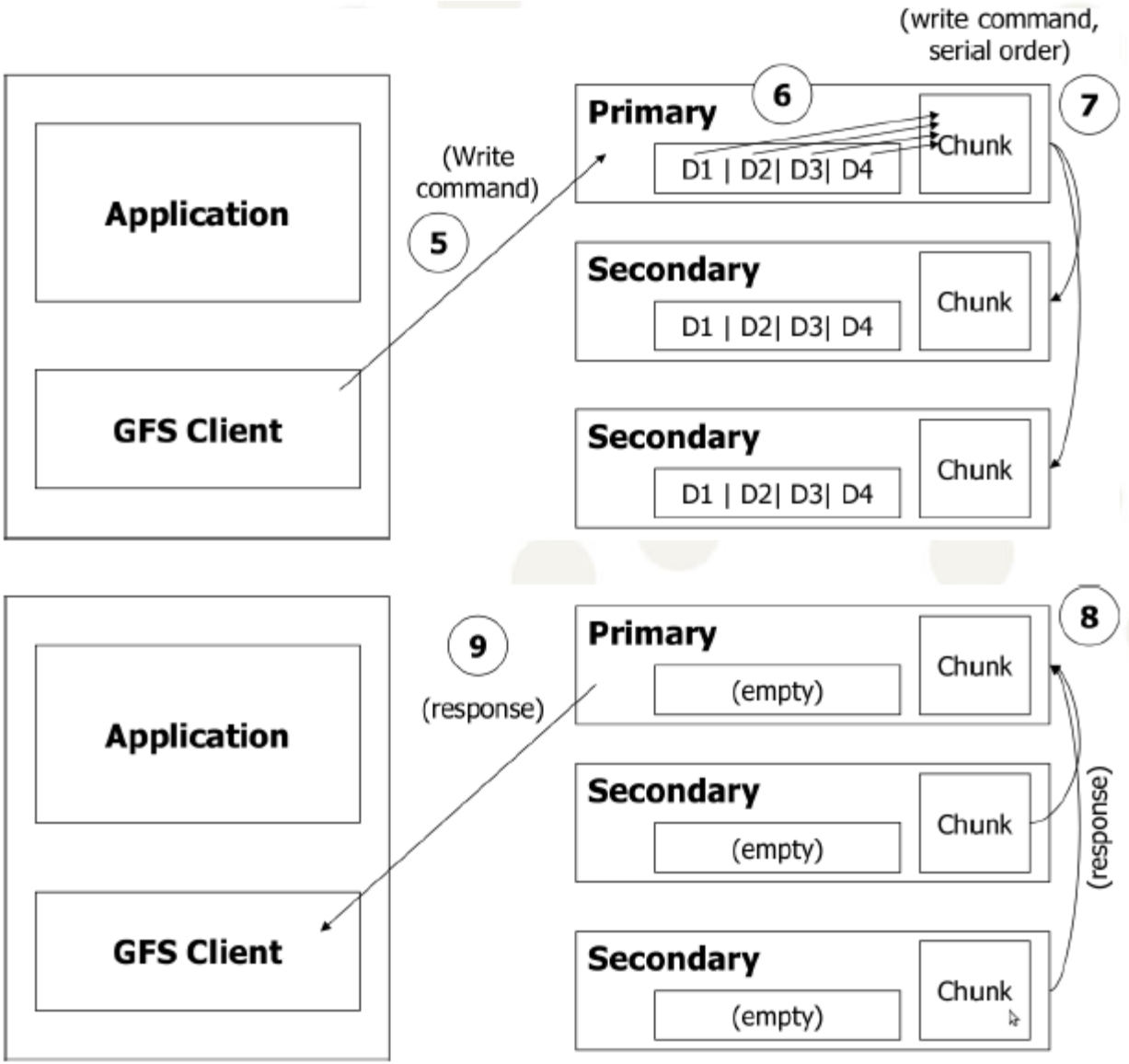
GFS : Read Operation



GFS : Write Operation (1/2)



GFS : Write Operation (2/2)



Google BigTable

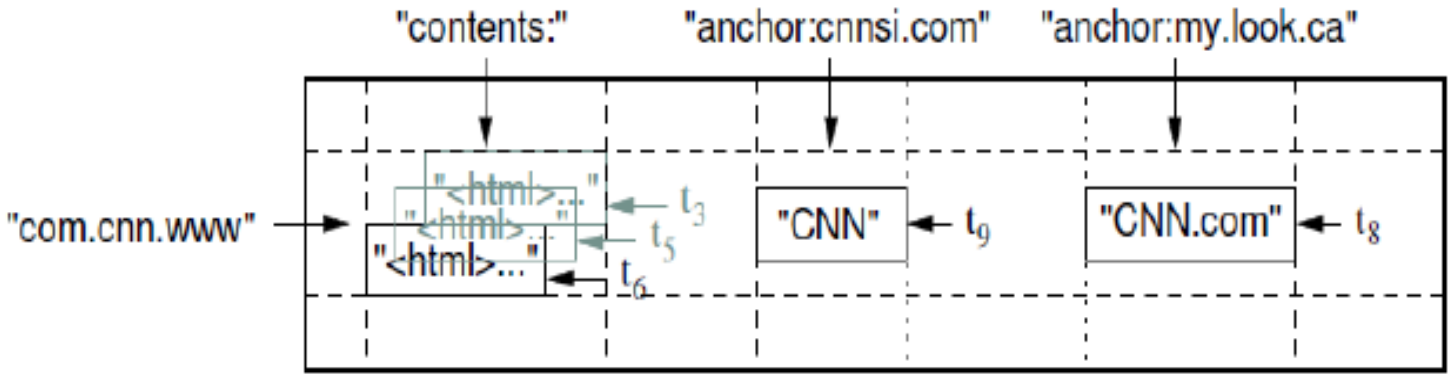
- BigTable is Google's distributed storage system for managing structured & unstructured data.
- Created in 2005, proprietary and maintained in-house
- Designed to scale to a very large size
  - Petabytes of data across thousands of commodity servers
  - Millions of user requests every second
- Used for many Google projects
  - Web indexing, Personalized Search, Google Earth, Google Analytics, Google Finance, … and more.
- Flexible, high-performance solution for all of Google's products

Google BigTable

- A Bigtable is a sparse, distributed, persistent multidimensional sorted map.
- The map is indexed by a row key, column key, and a timestamp.

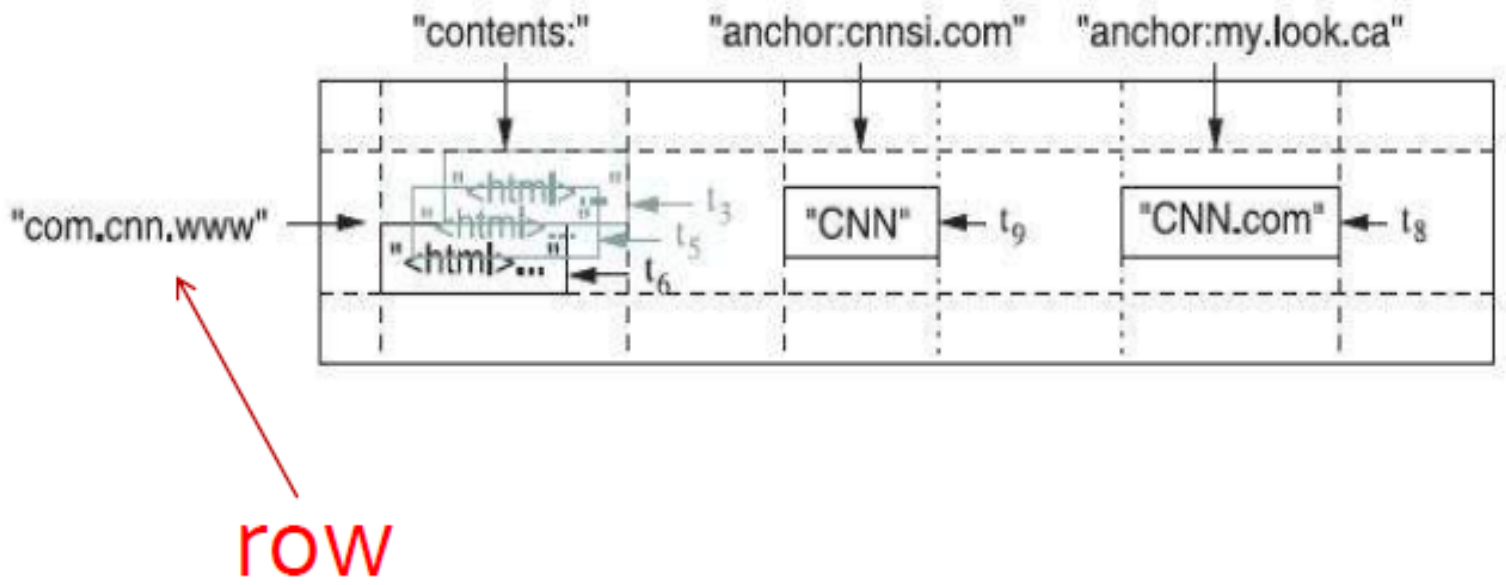
(row:string, column:string, time:int64) → string

Webtable



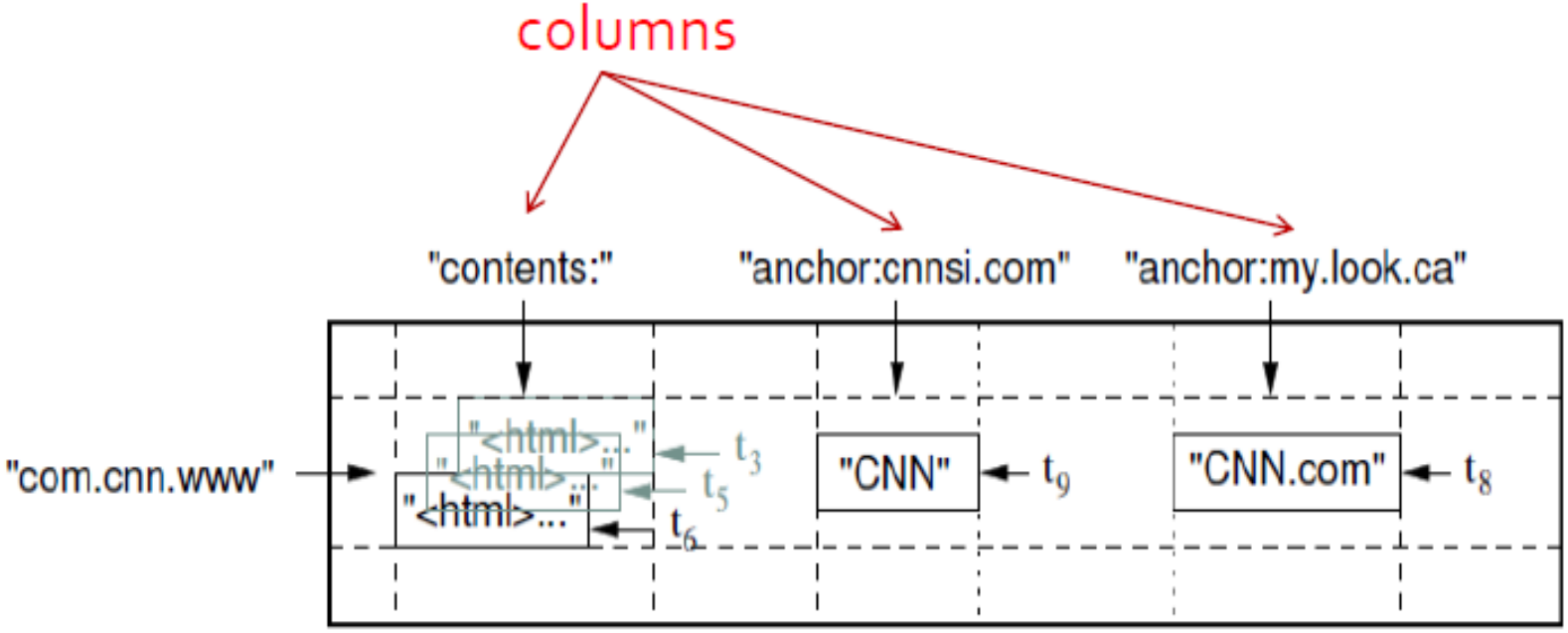
Google BigTable - Rows

- The row keys in a table are arbitrary strings.
- Data is maintained in lexicographic order by row key
- Each row range is called a tablet, which is the unit of distribution and load balancing.



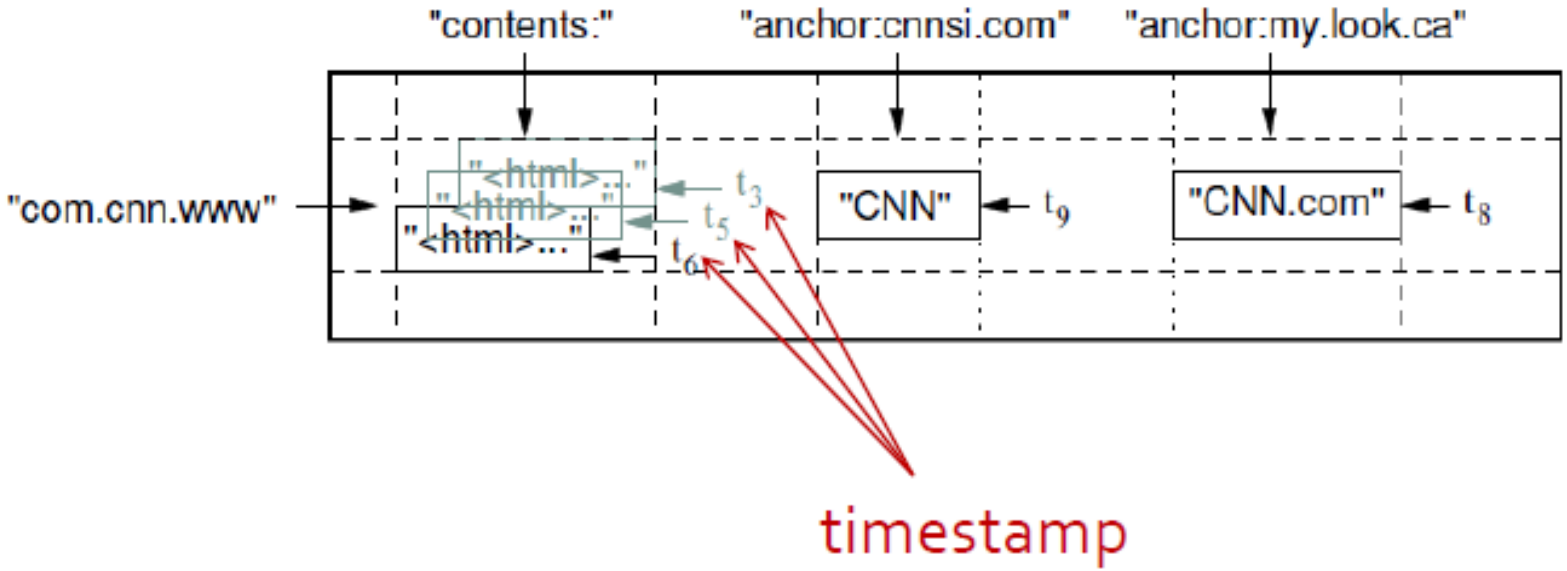
Google BigTable - Columns

- Column keys are grouped into sets called *column families*.
- Data stored in a column family is usually of the same type
- A column key is named using the syntax: *family* : *qualifier*.
- Column family names must be printable , but qualifiers may be arbitrary strings.

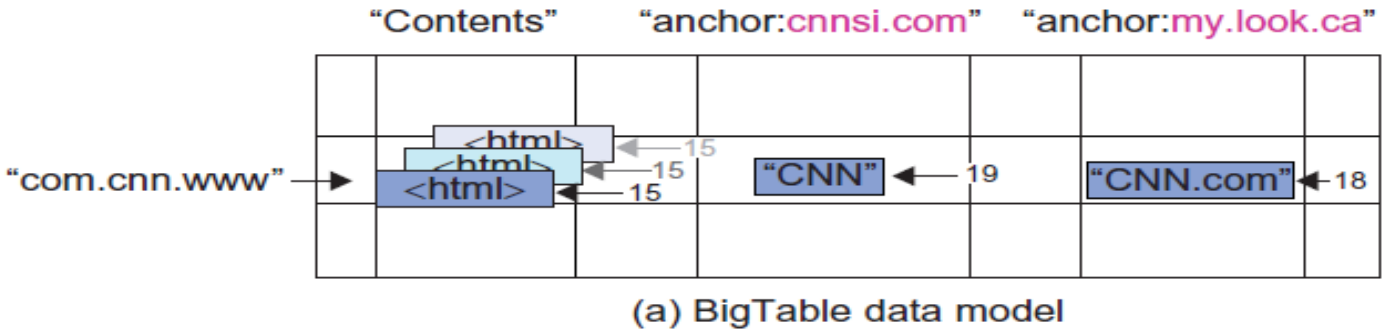


Google BigTable - Timestamps

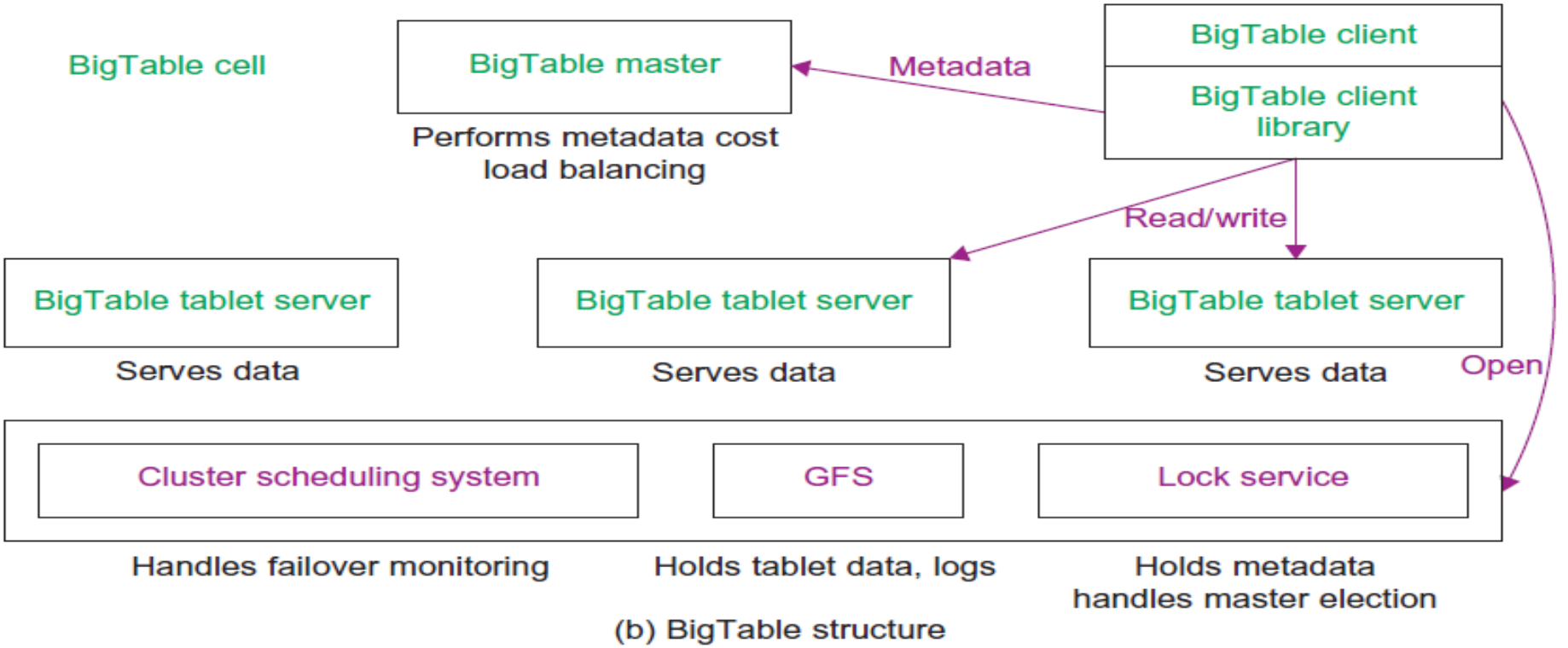
- Each cell in a Bigtable can contain multiple versions of the same data
- Versions are indexed by 64-bit integer timestamps
- Timestamps can be assigned:
  - automatically by Bigtable , or
  - explicitly by client applications



Google BigTable



(a) BigTable data model



(b) BigTable structure

Google BigTable

- Three major components
  - **Library linked into every client**
  - **Single master server**
    - Assigning tablets to tablet servers
    - Detecting addition and expiration of tablet servers
    - Balancing tablet-server load
    - Garbage collection files in GFS
  - **Many tablet servers**
    - Manages a set of tablets
    - Tablet servers handle read and write requests to its table
    - Splits tablets that have grown too large

Google BigTable

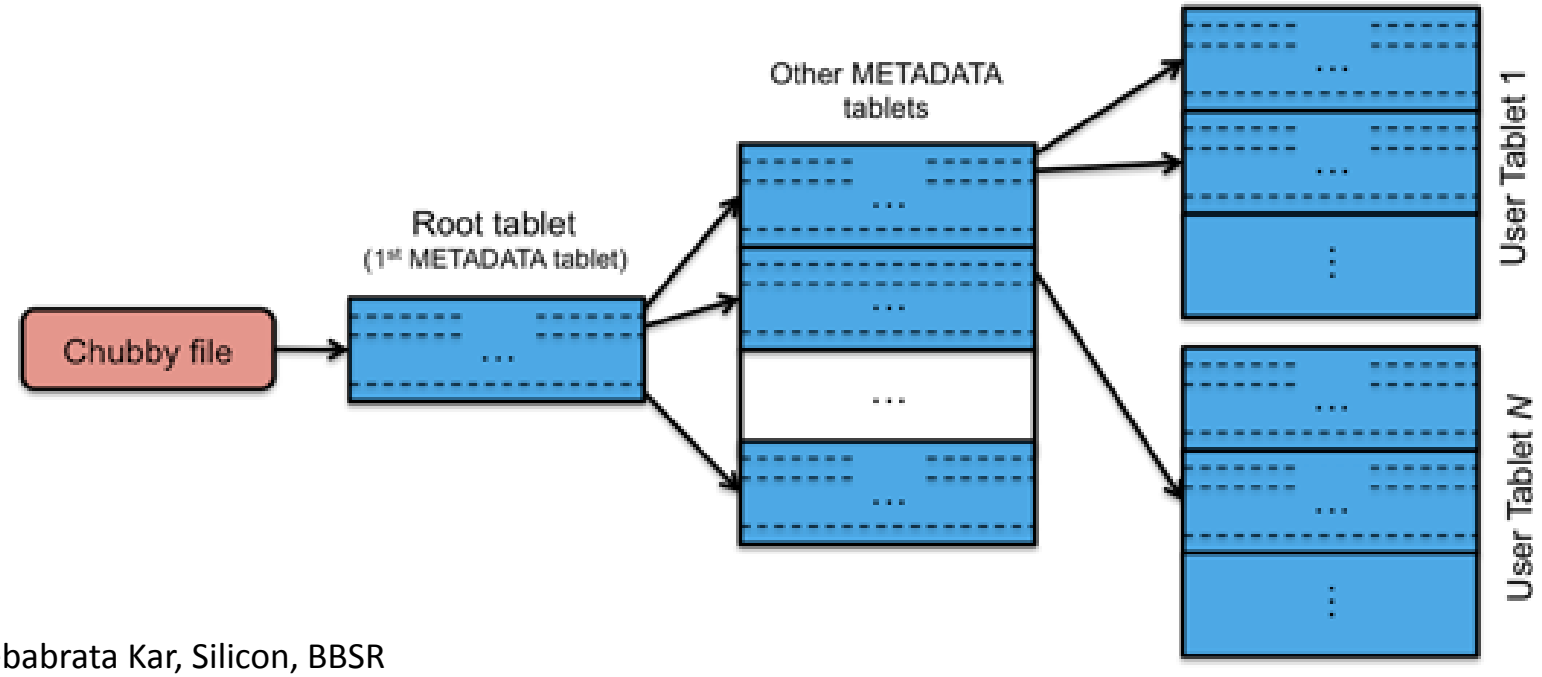
- Clients communicates directly with tablet servers for read/write

- Each table consists of a set of tablets
  - Initially, each table have just one tablet
  - Tablets are automatically split as the table grows

- Row size can be arbitrary (hundreds of GB)

Google BigTable – Locating Tablet

- Three level hierarchy
  - Level 2: Root tablet contains the location of METADATA tablets
  - Level 3: Each METADATA tablet contains the location of user tablets
  - Level 1: Chubby file containing location of the root tablet
    - Location of tablet is stored under a row key that encodes table identifier and its end row



Windows Azure

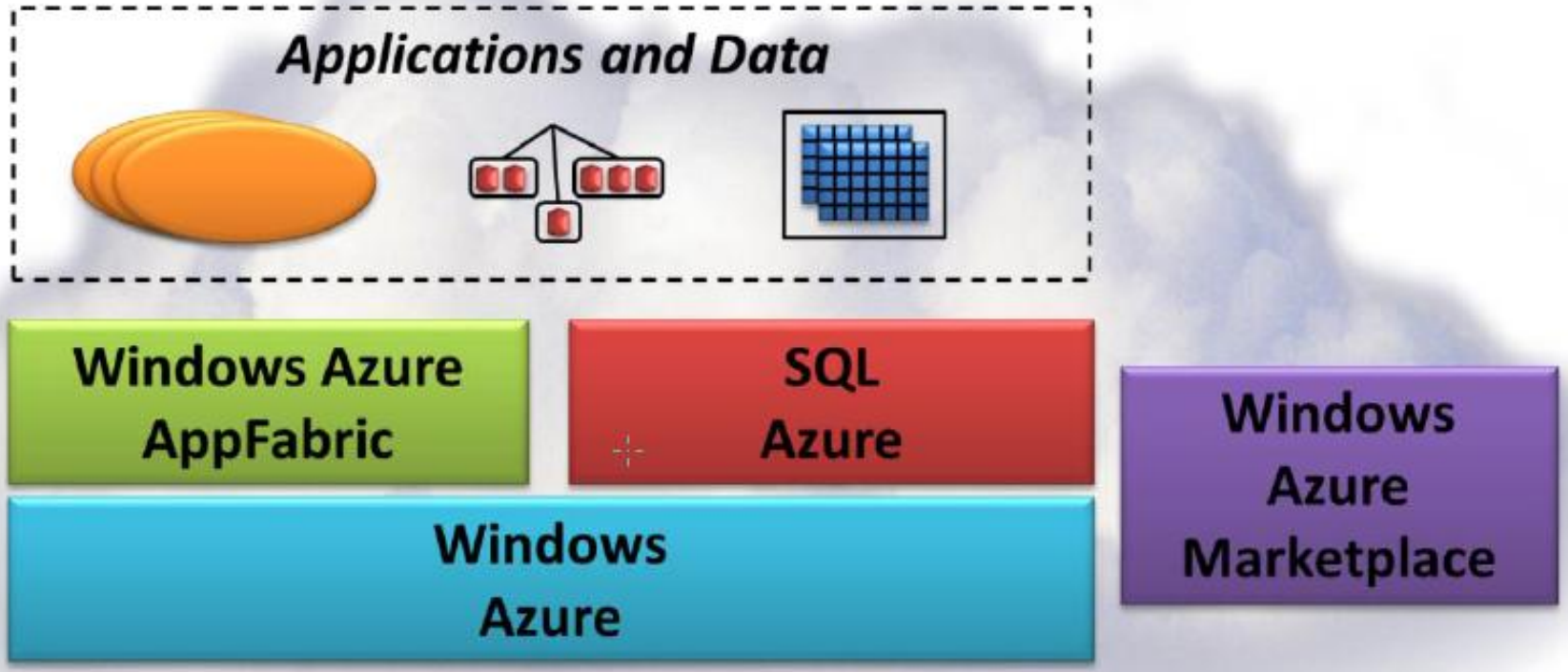
Renamed as
"Microsoft Azure" in 2014

Dr. Debabrata Kar
Silicon Institute of Technology, Bhubaneswar

Overview of Azure (PaaS)

- Microsoft's Cloud Computing Platform for building, testing, deploying, and managing applications and services on Microsoft-managed data centers.
- Platform-as-a-Service for running custom applications on pre-configured virtual machines.
- Supports wide range of app dev technology:
  - .NET Framework, Unmanaged code, others…
  - C#, VB, C++, Java, ASP.NET, WCF, PHP, Python etc.
- Provides several storage options
  - For simple data structures as well as BLOBS
  - Also, traditional Relational Databases through SQL Azure
- Connectivity with other distributed applications
  - Along with on-premises applications through firewalls

Overview - Windows Azure Platform



Overview - Windows Azure Platform

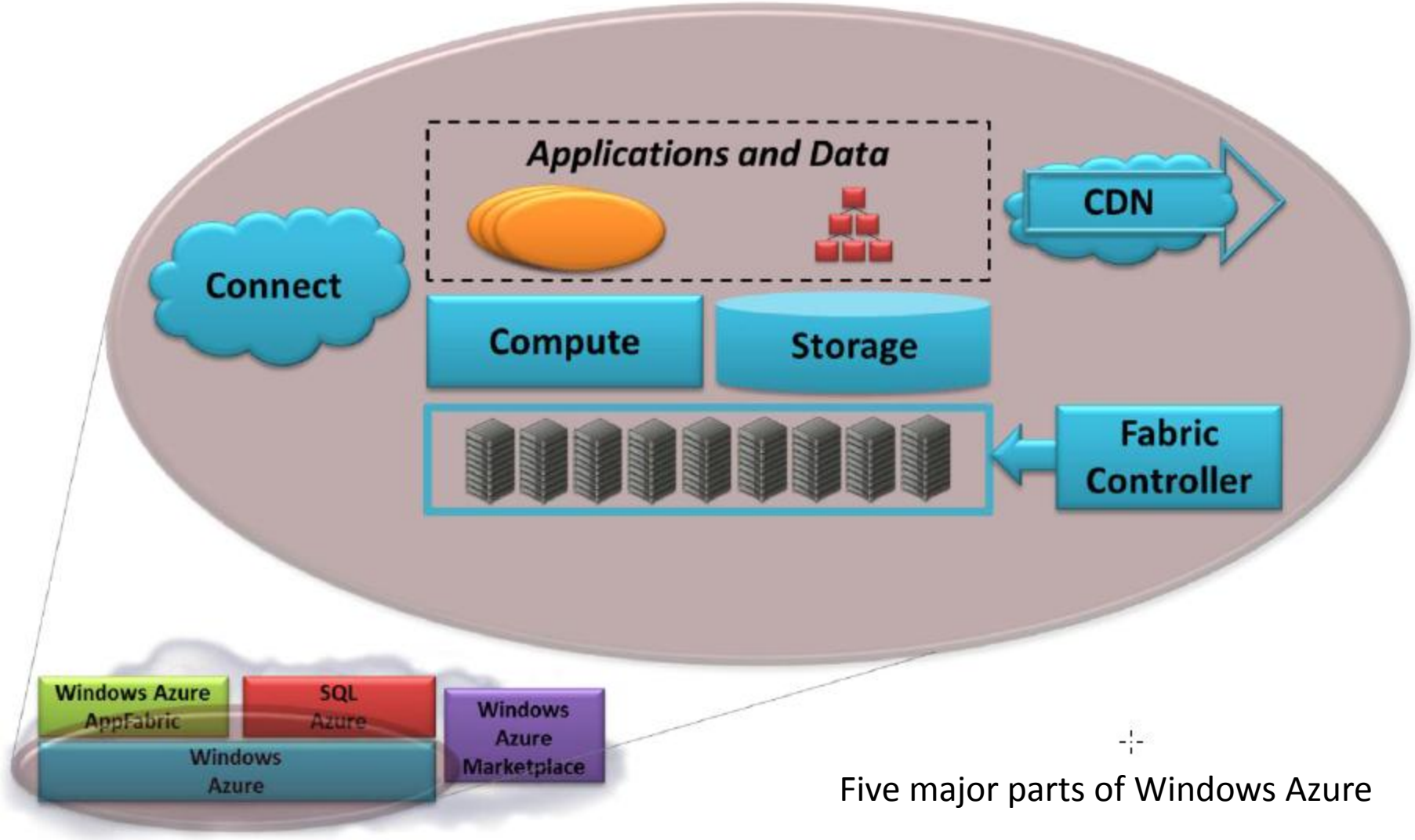
Consists of the following 4 major components:

- **Windows Azure**: A Windows environment for running applications and storing data on computers in data centers managed by Microsoft.
- **SQL Azure**: Relational database services in the cloud based on Microsoft SQL Server.
- **Windows Azure AppFabric**: Cloud-based infrastructure services for running apps in the cloud or on premises.
- **Windows Azure Marketplace**: An online service for purchasing cloud-based data and applications.

All four of these components run in Microsoft data centers located around the world (54 Regions, 140 Countries)

Windows Azure

- Runs windows applications and stores data on the cloud.



Five major parts of Windows Azure

1. Windows Azure - Compute

- The Windows Azure compute service runs applications on a Windows Server foundation.
- Applications can be created using the .NET Framework in C#, Visual Basic, C++, Java, and other languages.
- Developers can use Visual Studio for developing & deploying applications, or use other development tools.
- Technologies such as ASP.NET, Windows Communication Foundation (WCF), and PHP can also be used.

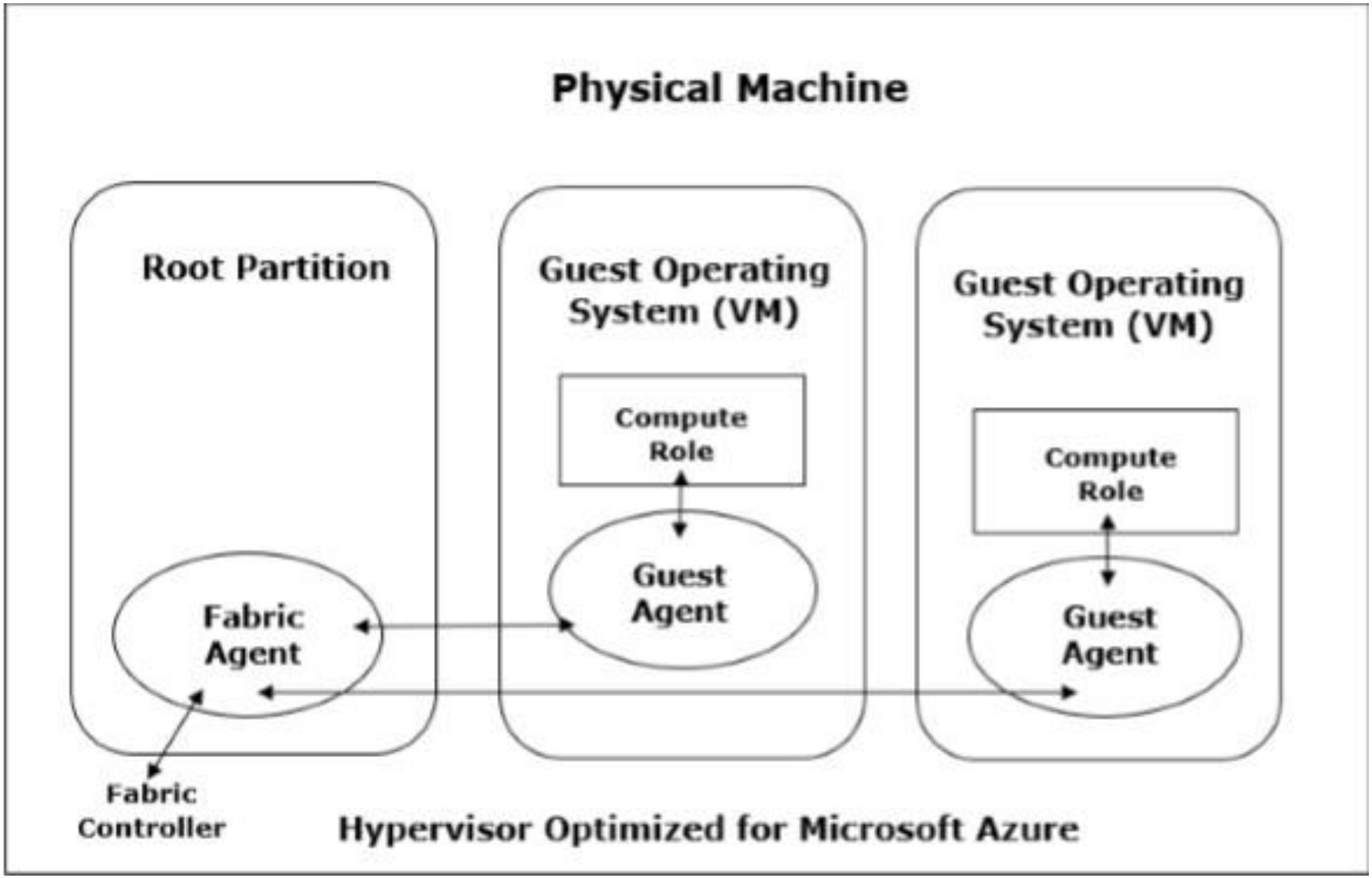
2. Windows Azure - Storage

- Azure storage service allows storing simple data structures or binary large objects (BLOBs)
- It provides queues for communication between components of Windows Azure applications
- Also offers a form of tables with a simple query language.
- Windows Azure applications that need traditional relational storage can also use SQL Azure.
- Both Windows Azure applications and on-premises applications can access the Windows Azure storage service, and both do it in the same way.
  - using a RESTful approach.

3. Windows Azure - Fabric Controller

- Windows Azure runs on a large number of machines in the data centers around the world.
- The Fabric Controller's job is to knit the machines in a Azure DC into a cohesive pool of system resources.
  - Basically, a proprietary clustering system by Microsoft
  - It functions as the kernel of the Azure operating system.
  - The kernel consists of a Fabric Agent (hypervisor)
  - It provisions, stores, delivers, monitors and commands the VMs and other physical servers that make up the Azure platform.
- The Windows Azure compute and storage services are built on top of this pool of compute and storage resources provided by Fabric Controller.

3. Windows Azure - Fabric Controller



4. Windows Azure - CDN

- The Azure Content Delivery Network (CDN) is a global CDN solution for delivering high-bandwidth content.
- Can be hosted in Azure or any other location.
- Can perform caching frequently accessed data
  - Static objects loaded from Azure Blob storage, a web application, or any publicly accessible web server.
- Caching is done considering proximity to the users (closest POP server) to provide faster access to the data.
- The Windows Azure CDN can also do caching for BLOBs, maintaining cached copies at sites around the world depending on the users' needs.

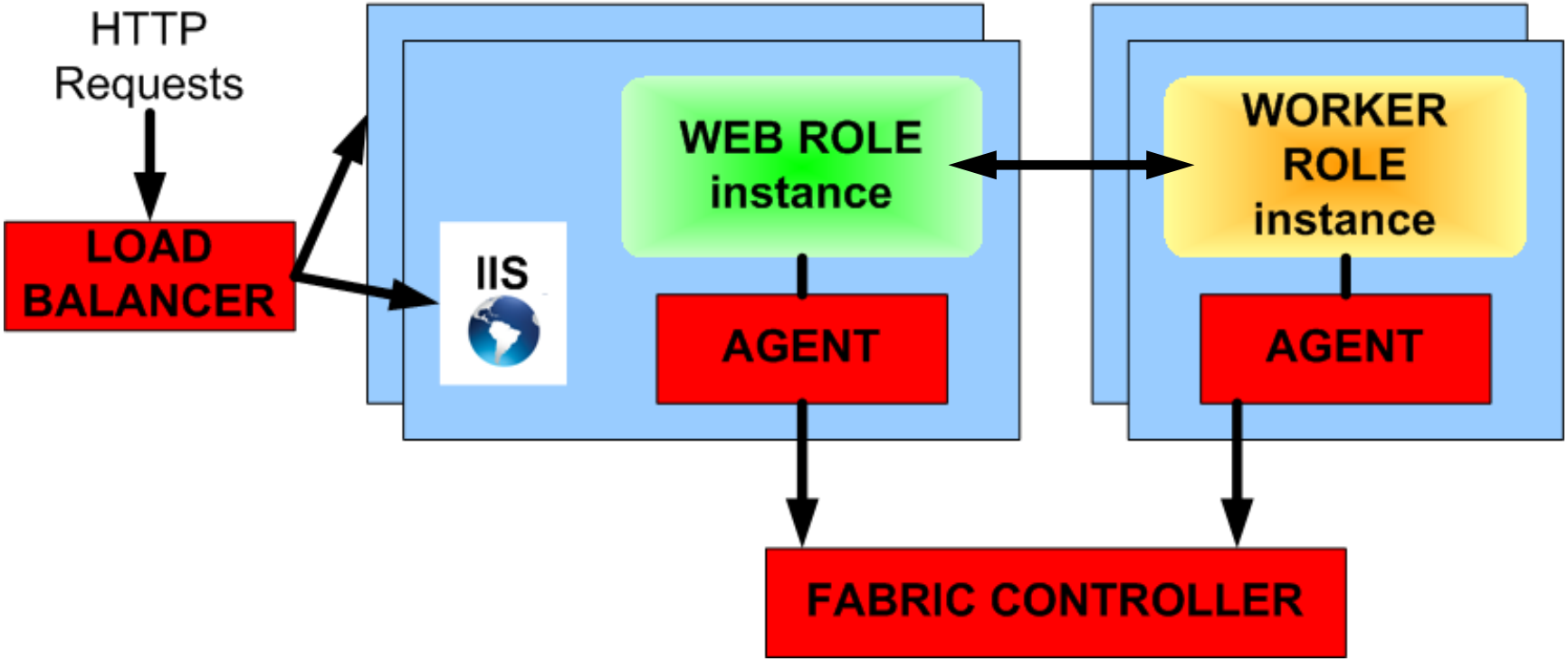
5. Windows Azure - Connect

- Provides connectivity components for applications deployed on Windows Azure platform.
- Allows organizations to interact with cloud applications as if they were inside the organization's own firewall.
- It takes care of all the operations that are related to synchronize *identity data* between the on-premises application environment and Azure Active Directory

Azure Applications

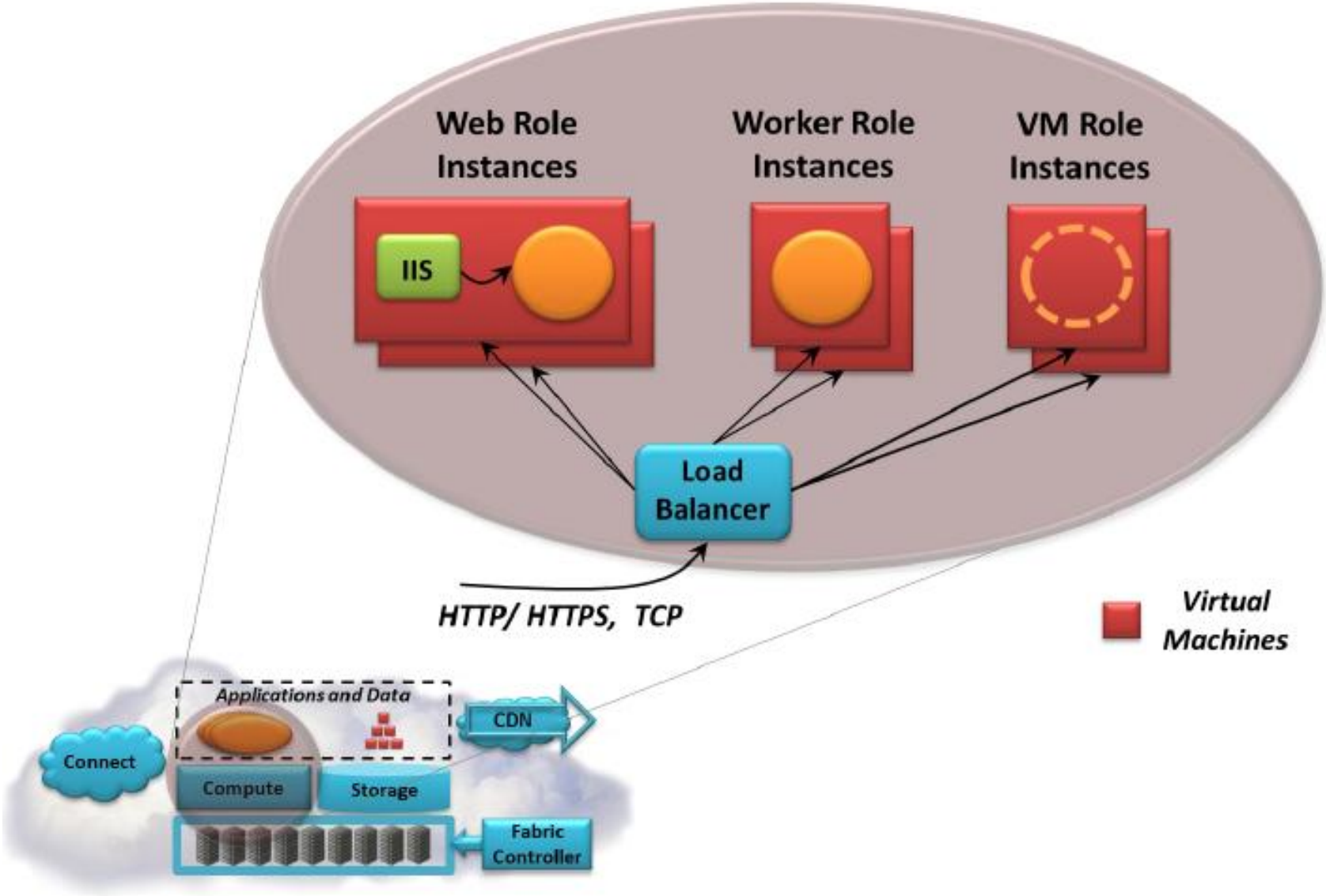
- Azure applications hosted on the Azure cloud platform run in one of the following two roles:
  - **Web Role**
    - Primarily for running Web-based applications
    - It is the Front-End of Azure hosted applications
    - Can accept HTTP/HTTPS requests on standard ports
    - It is automatically deploys and hosts the application through IIS
    - Supports ASP.NET, FastCGI + PHP etc.
  - **Worker Role**
    - Designed to run a variety of code for supporting services of web roles
    - It is the Back-End of Azure hosted applications
    - Can NOT accept HTTP/HTTPS requests - uses other TCP ports
    - Is not hosted on IIS, but as standalone applications/services
- Developer can specify how many instances to run

Azure Applications



- Agent
  - Exposes the API
  - Monitors the failure conditions of the application
- Fabric Controller
  - Allocate resources according to configuration file
  - Detect and restart failed web roles and workers

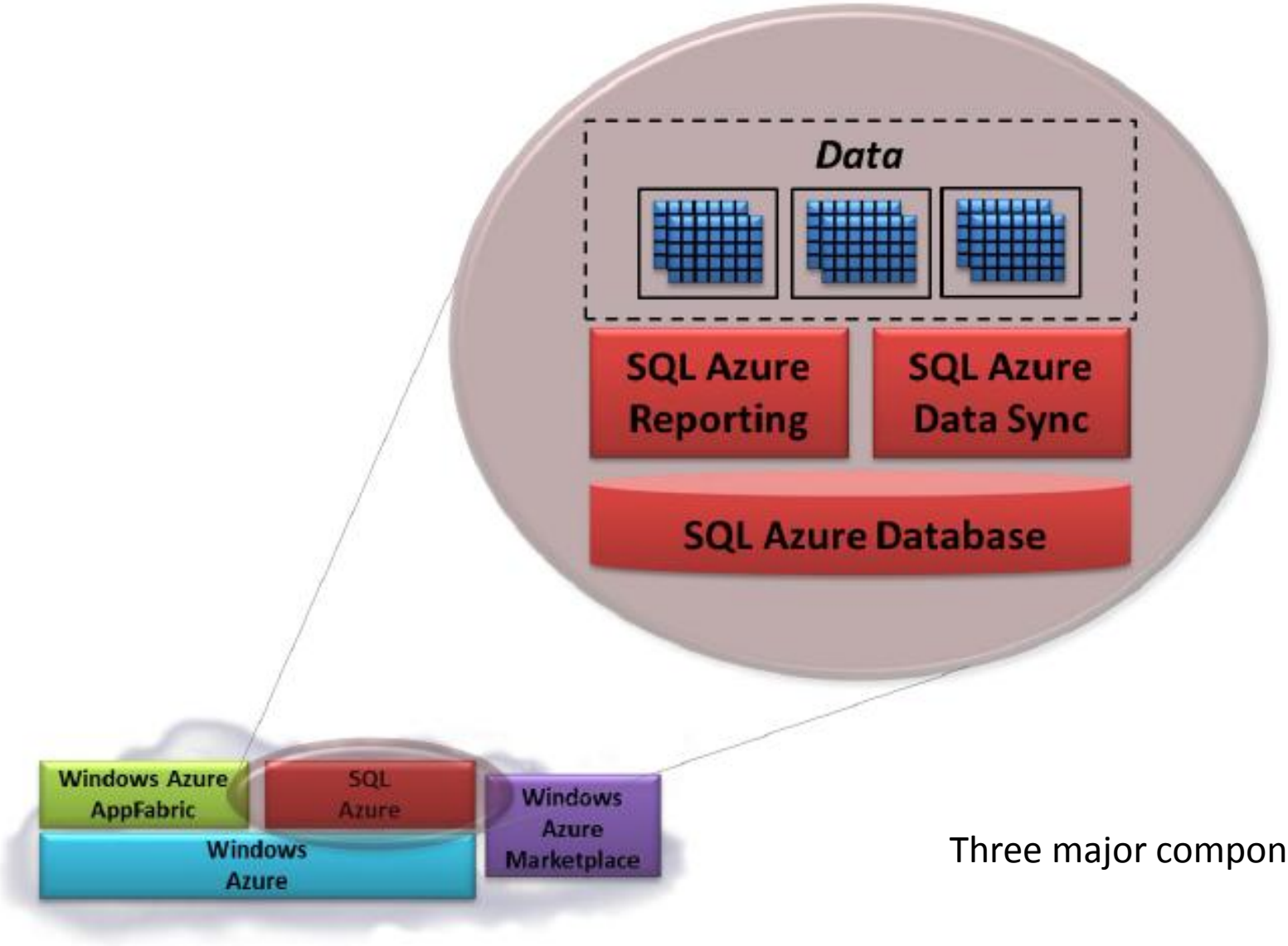
Azure Applications



SQL Azure

- SQL Azure offers cloud-based services for relational data.
- On-demand provisioning of Microsoft SQL Databases.
- Based on SQL Server 2008 R2 engine.
- Familiar relational database server
  - Same SQL query language → consistent development model
  - Makes migrating SQL databases to the cloud easy
- Uses same tools and data access frameworks
- Each user account can have multiple logical servers
- Each server can have multiple databases
  - Database can scale up to 50 GB by default
  - Snapshot can be taken (for backup purpose)

SQL Azure

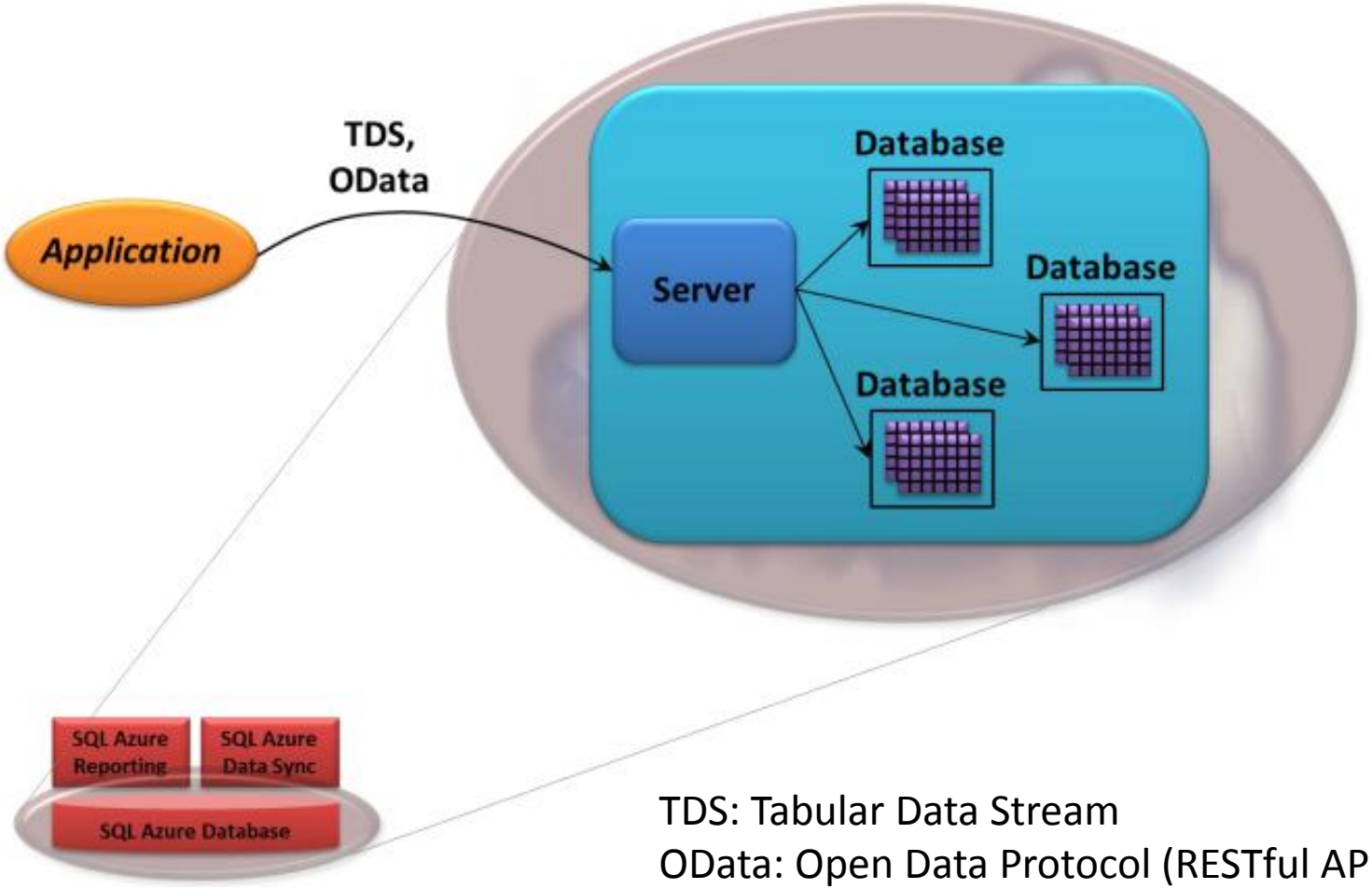


Three major components

SQL Azure

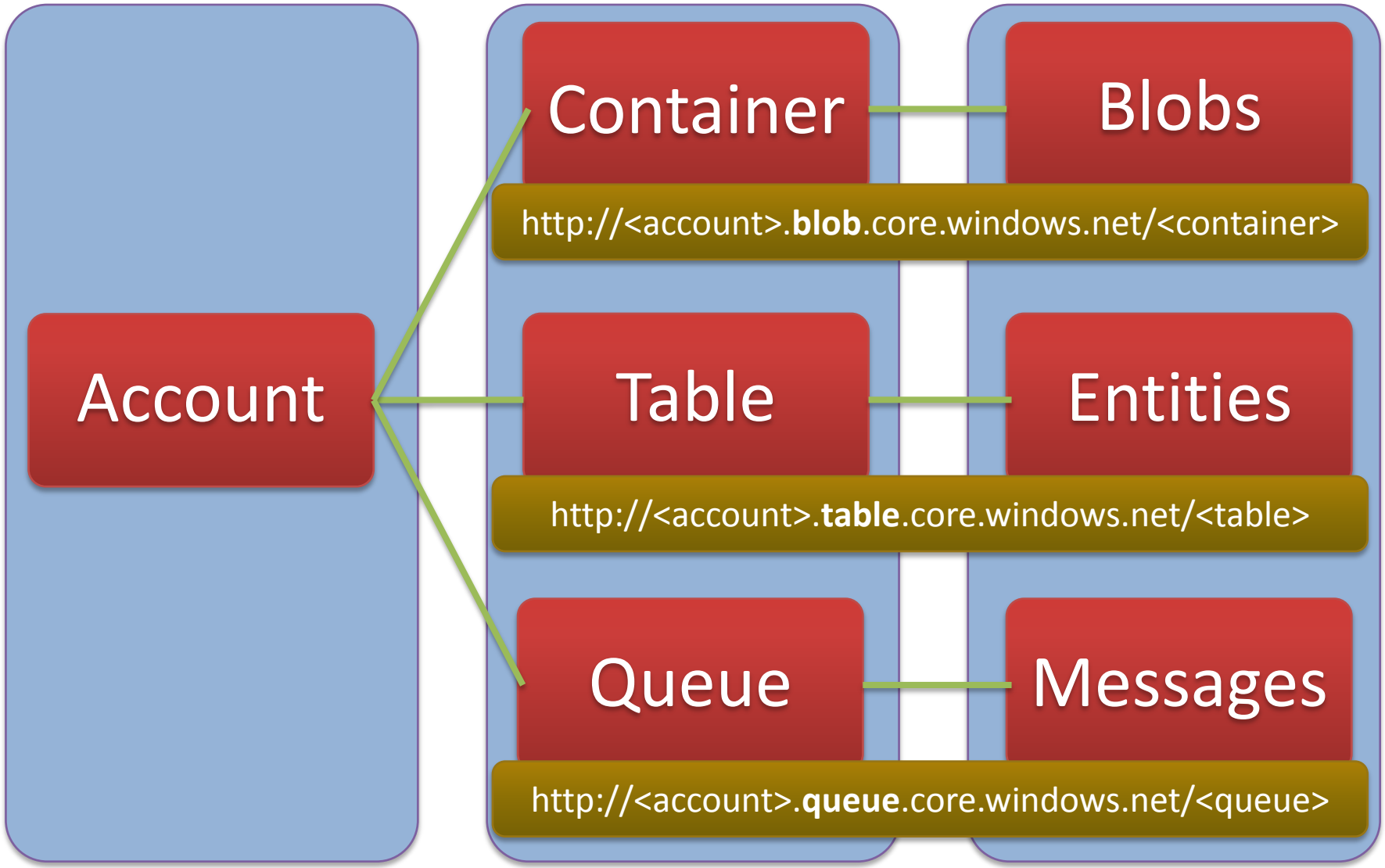
- **SQL Azure Database:**
  - Provides a cloud-based database management system (DBMS)
  - Allows on-premises and cloud applications to store relational data on SQL Servers managed by Microsoft.
- **SQL Azure Reporting:**
  - It is a version of SQL Server Reporting Services (SSRS) that runs in the cloud (intended primarily for use with SQL Azure)
  - It allows creating and publishing SSRS reports on cloud data.
- **SQL Azure Data Sync:**
  - Allows synchronizing data between SQL Azure Database and on-premises SQL Server databases.
  - Can also be used to synchronize data across different SQL Azure databases in different Microsoft data centers.

SQL Azure Database



TDS: Tabular Data Stream
OData: Open Data Protocol (RESTful API)

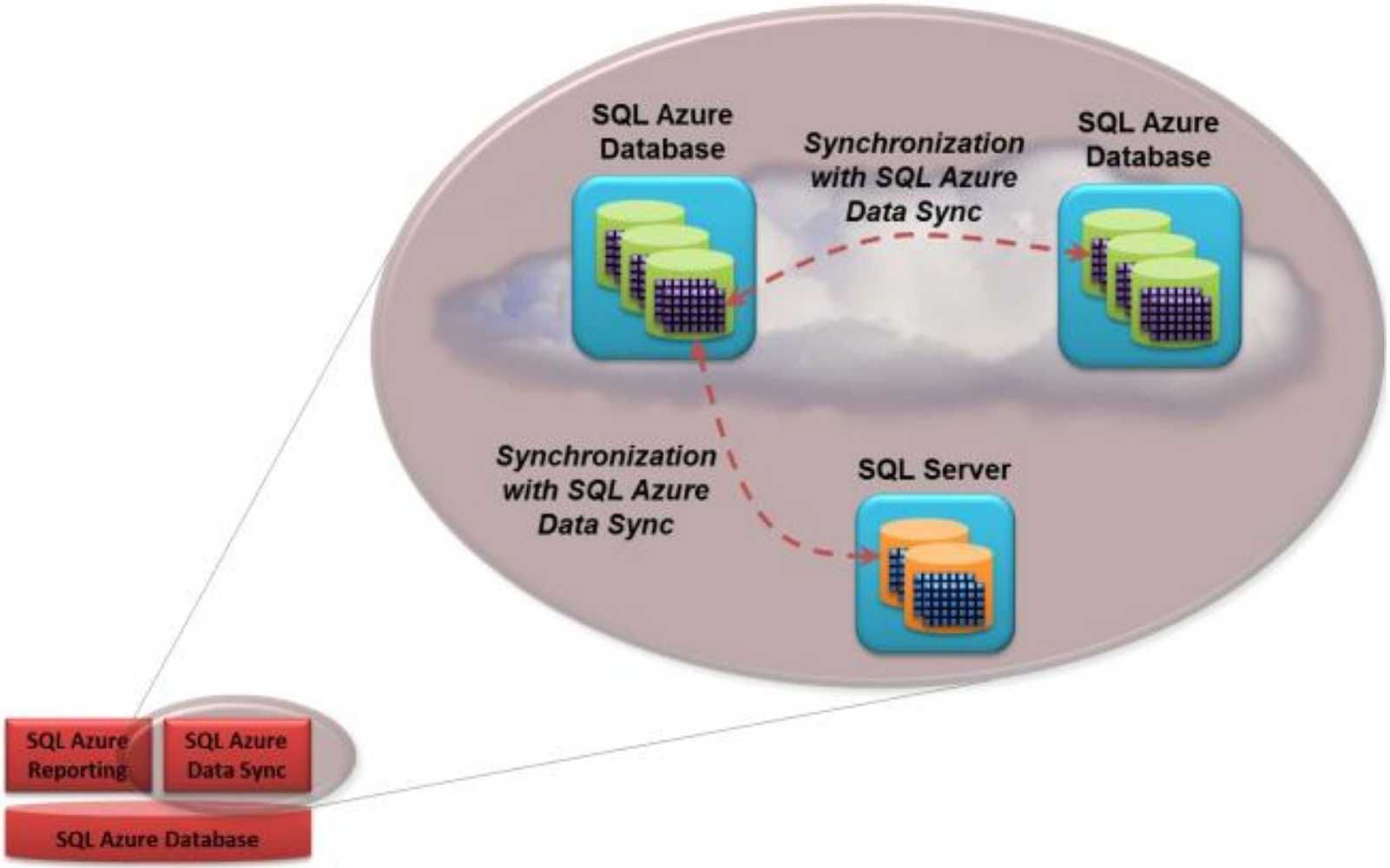
Windows Azure Data Storage



SQL Azure Reporting

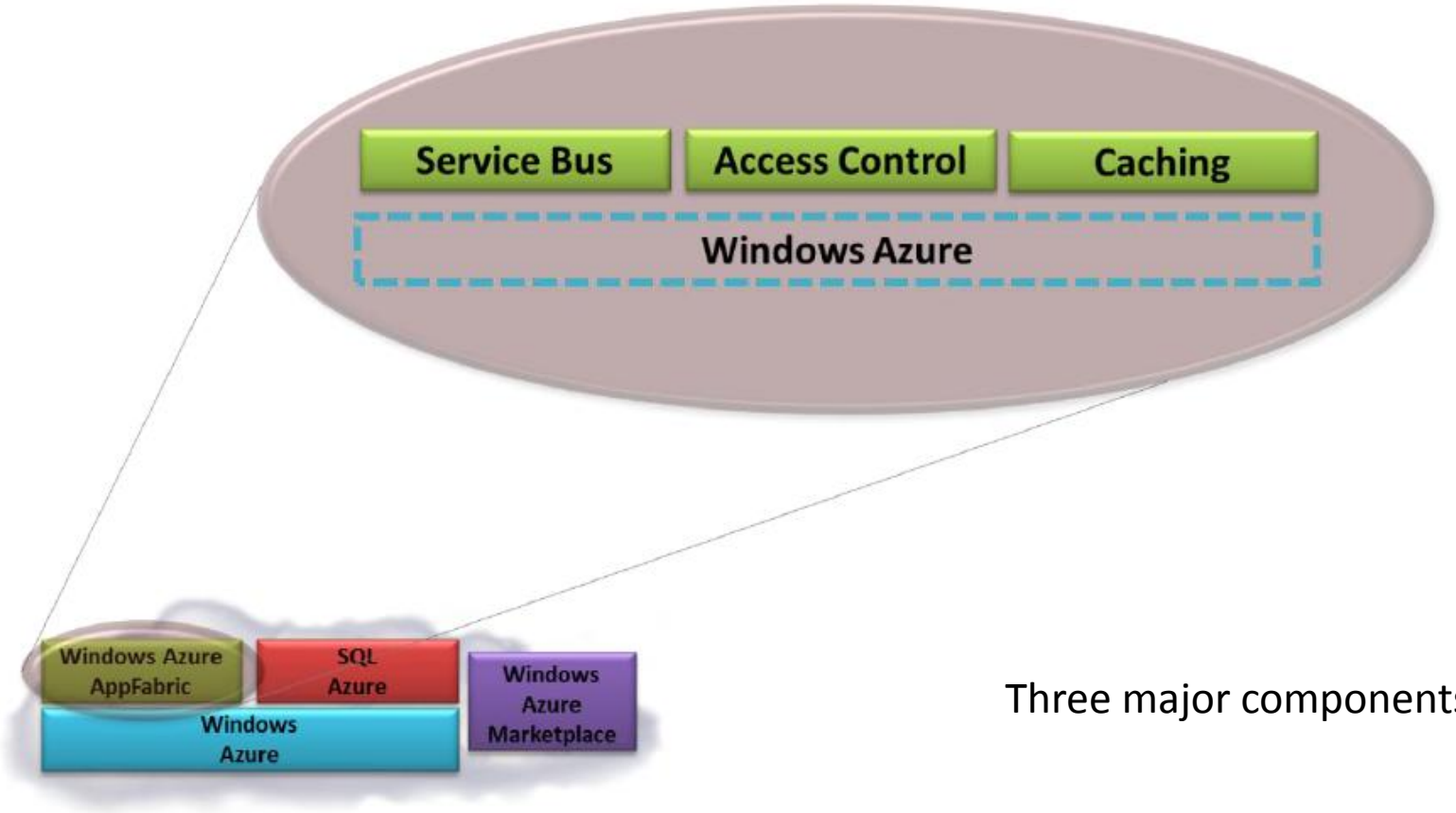
- Based on MS SQL Server Reporting Services (SSRS)
- Provides a cloud-based way to create reports quickly.
- Reports created using SQL Azure Reporting can be published to a SQL Azure Reporting portal, accessible directly via a URL.
- Developers can embed reports published to the SQL Azure Reporting portal in Windows Azure applications.

SQL Azure – Data Sync



Windows Azure AppFabric

- AppFabric addresses common challenges for building distributed applications on the cloud.

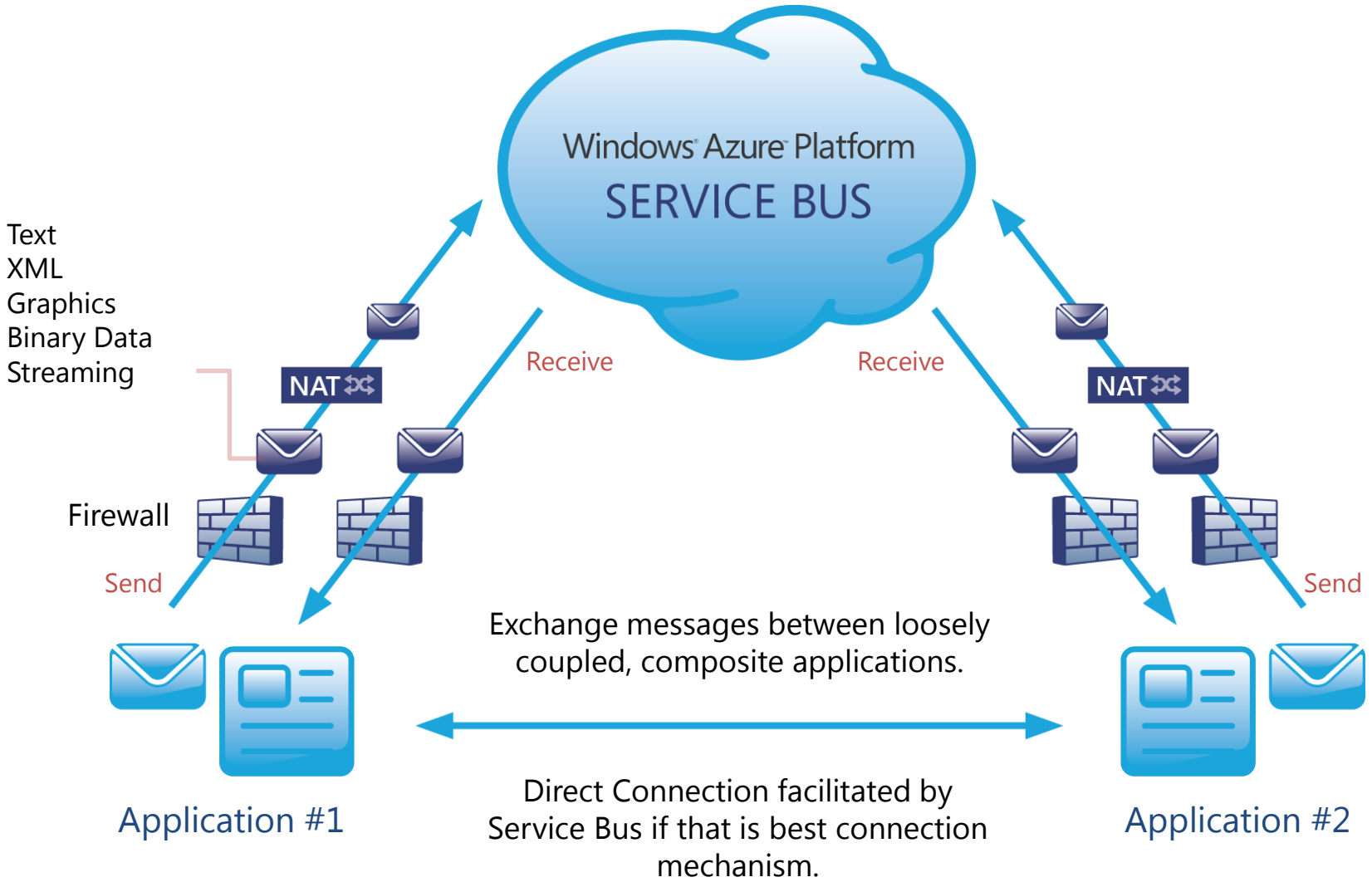


Three major components

Windows Azure AppFabric

1. **Service Bus**:
   - General purpose application bus and connectivity service.
   - Makes it simpler by allowing applications to expose endpoints in the cloud that can be accessed by other applications (on-premises or in the cloud).
   - Each exposed endpoint is automatically assigned a URI, which can be used by clients to locate and access the service.
   - Service Bus also handles the challenges of dealing with network address translation (NAT) and getting through firewalls without opening new ports for exposed applications.

AppFabric – Service Bus

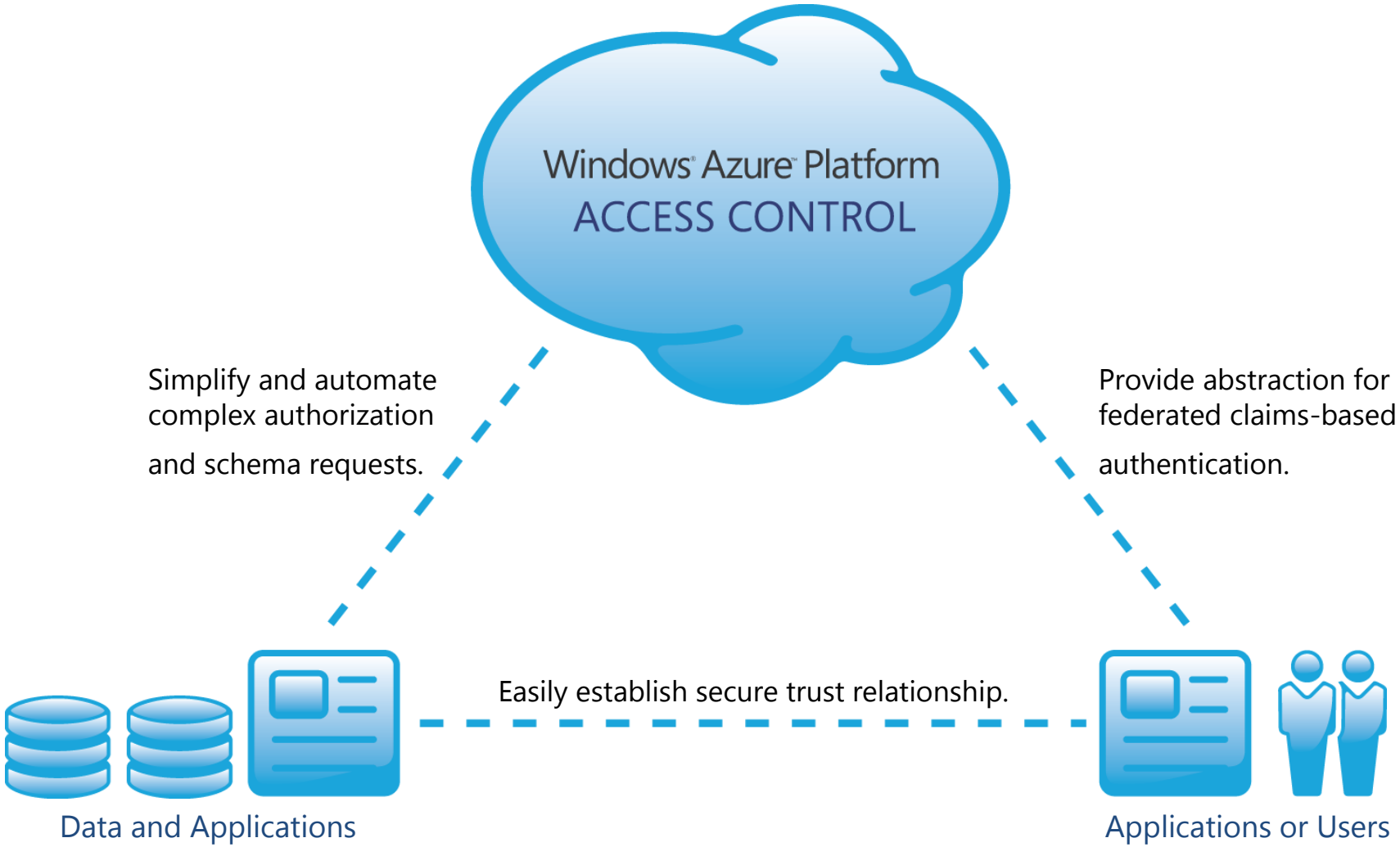


Windows Azure AppFabric

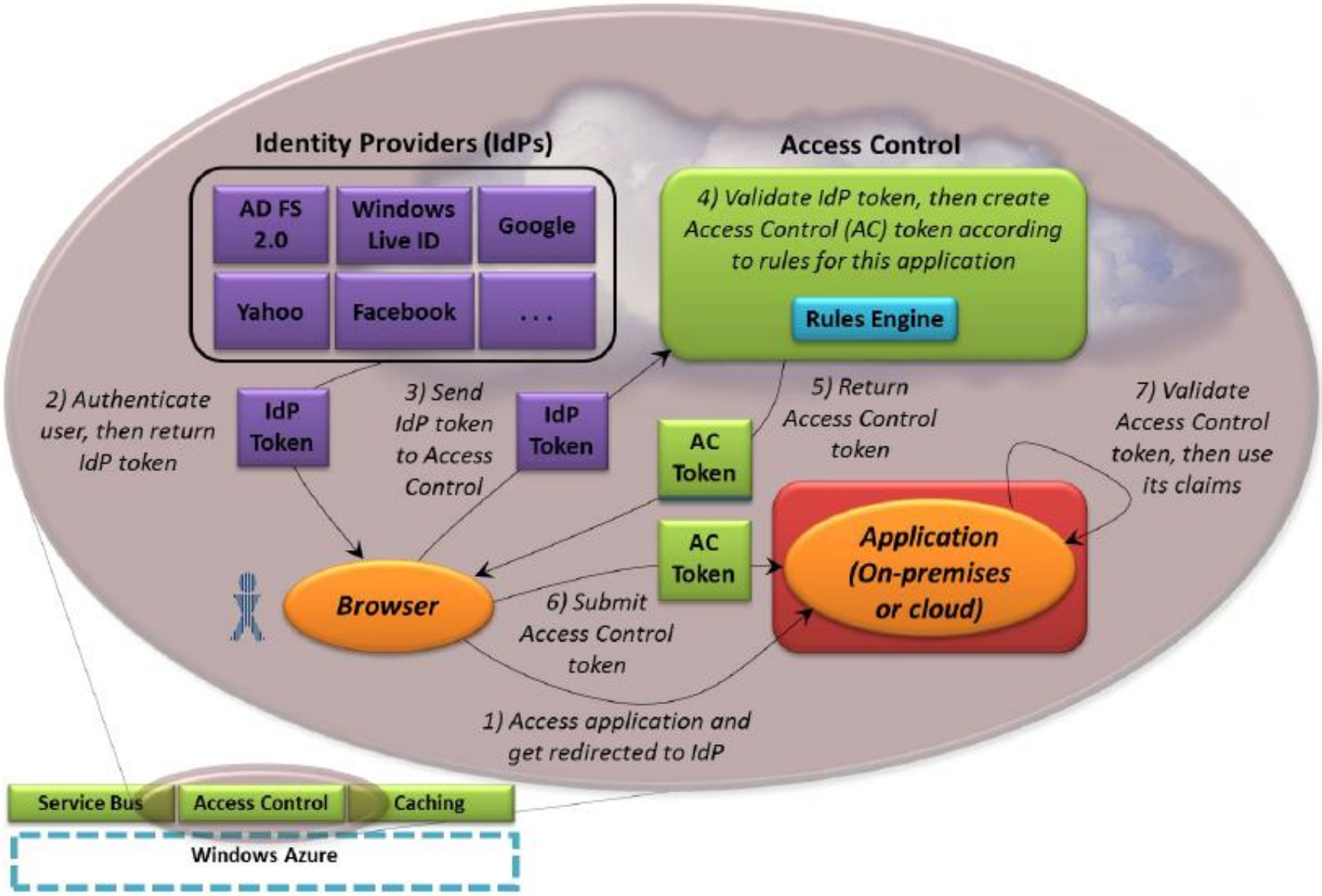
2. **Access Control**:

- Rules-driven, claims-based access control service.
- Digital identity can include Active Directory, Windows Live ID, Google Accounts, Facebook, and many more …
- Letting users log in with any of these is a daunting task for the application developers.
- The Azure AppFabric Access Control component simplifies this by providing built-in support for all of them.
- Also provides a single place for defining rules to control what each user is allowed to access.

AppFabric – Access Control



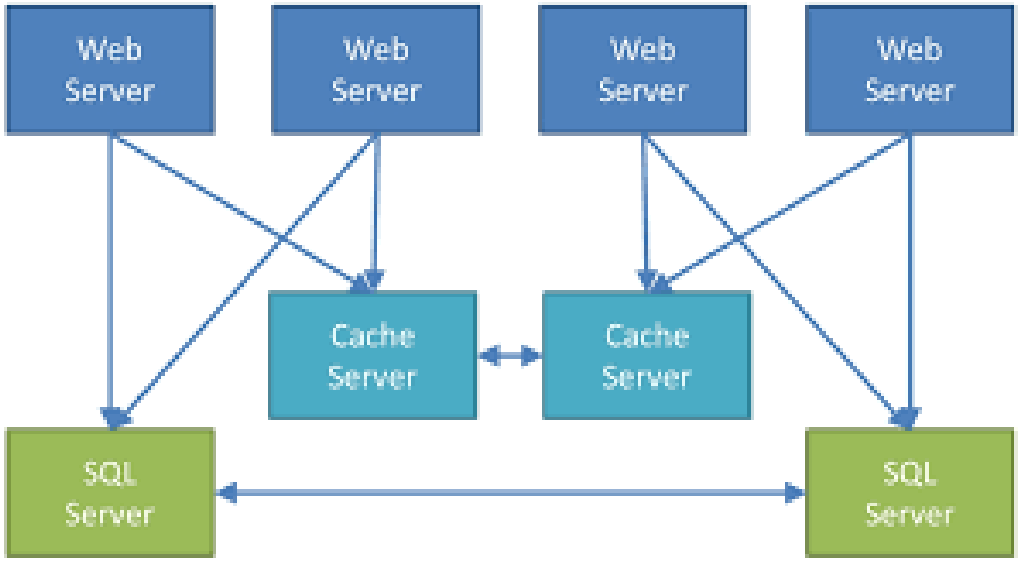
AppFabric – Access Control



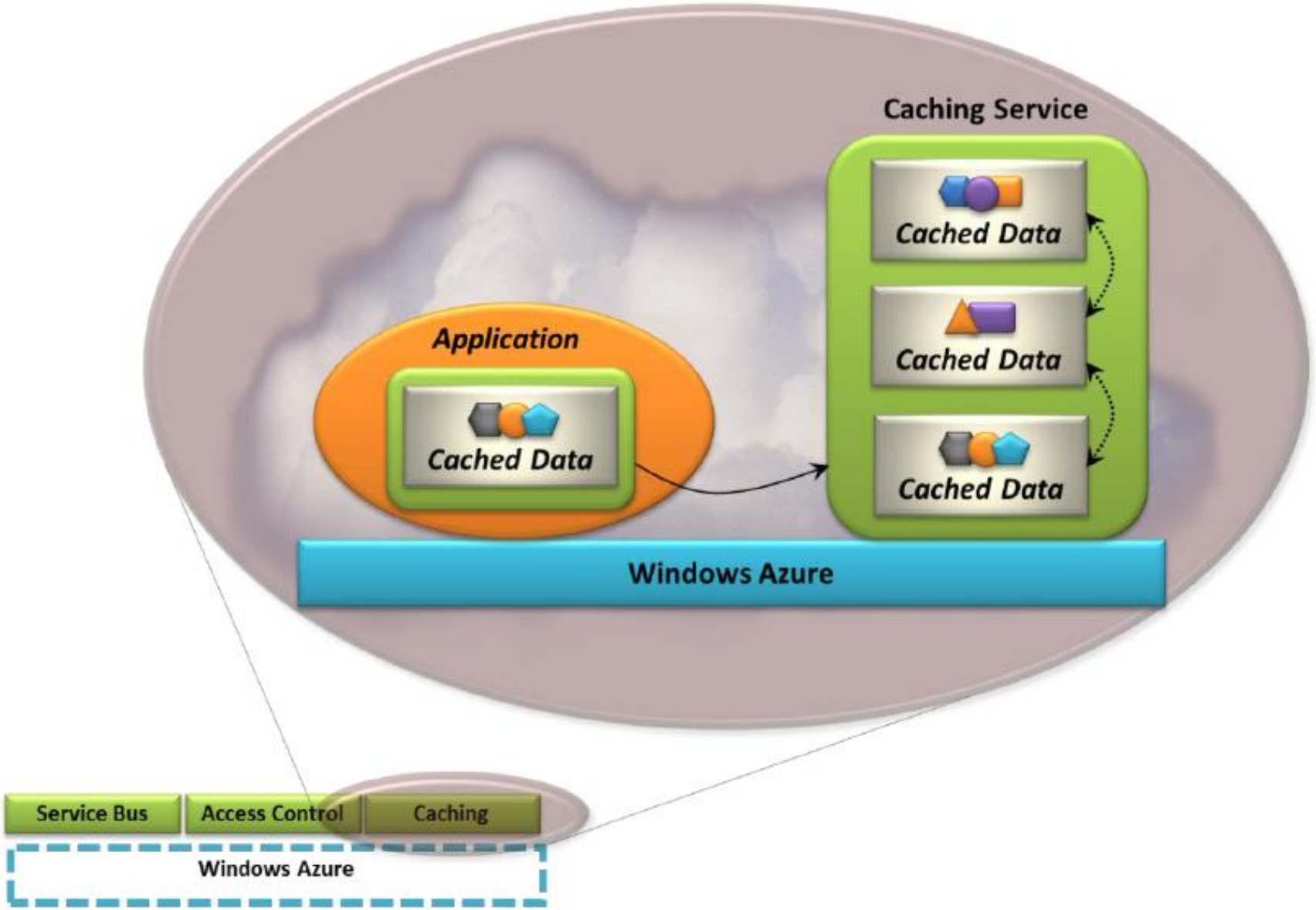
Windows Azure AppFabric

3. Caching:

- Caching of frequently accessed information must be done to speed up these kind of applications
- Caching reduces the number of times an application issues a query on a database, by providing the cached results from previous query.
- The AppFabric Caching service provides in-memory caching service to boost performance of Windows Azure applications.



Windows Azure AppFabric - Caching



Windows Azure Platform Consumption Prices

Pay as you go and grow for only what you use when you use it

Windows Azure

Elastic, scalable, secure, and highly available automated service platform

Compute

Per service hour

$0.12/hour
+ Variable Instance Sizes

Storage

Per GB stored and transactions
$0.15 GB/month
$0.01/10k transactions

Microsoft SQL Azure

Highly available, scalable, and self managed distributed database service

Web Edition

Per database/month

$9.99/month
(up to 1 GB DB/month)

Business Edition

Per database/month

Starting at $99.99/month
(10-50 GB DB/month)

Windows Azure AppFabric Service Bus and Access Control Service

Scalable, automated, highly available services for secure connectivity

Access Control

Per Message Operation

$1.99/10k transactions

Service Bus

Per Message Operation

$3.99/month per connection

Windows Azure Instance Sizes

Variable instance sizes to handle complex workloads of any size

Small	Medium	Large	X-Large
$0.12	$0.24	$0.48	$0.96
Per service hour	Per service hour	Per service hour	Per service hour

Unit of Compute Defined

Equivalent compute capacity of a 1.6 Ghz Processor (on 64-bit platform)

Small	Medium	Large	X-Large
1 x 1.6Ghz	2 x 1.6Ghz	4 x 1.6Ghz	8 x 1.6Ghz
(moderate IO)	(high IO)	(high IO)	(high IO)
1.75 GB memory	3.5 GB memory	7.0 GB memory	14 GB memory
250 GB storage (instance storage)	500 GB storage (instance storage)	1000 GB storage (instance storage)	2000 GB (instance storage)

SalesForce.com

World Leader in CRM

Dr. Debabrata Kar
Silicon Institute of Technology, BBSR

Overview of SalesForce.com (SaaS)

- SalesForce is the world's first and most popular CRM (Customer Relationship Management) software.
- It was founded by former Oracle Executive Marc Benioff and others in March 1999, HQ - San Francisco, CA.
- SalesForce is offered as a low cost, low risk, cloud based Software-as-a-Service (SaaS) solution in CRM domain.
- In 2014, SalesForce launched Customer Success Platform to tie together all SalesForce services
  - Sales, Service, Marketing, Analytics, Community, Mobile Apps
- About 150,000 companies use SalesForce
  - More than 3.75 million subscribers, $8.4 bn Revenue
- Has a multi-tenant architecture.

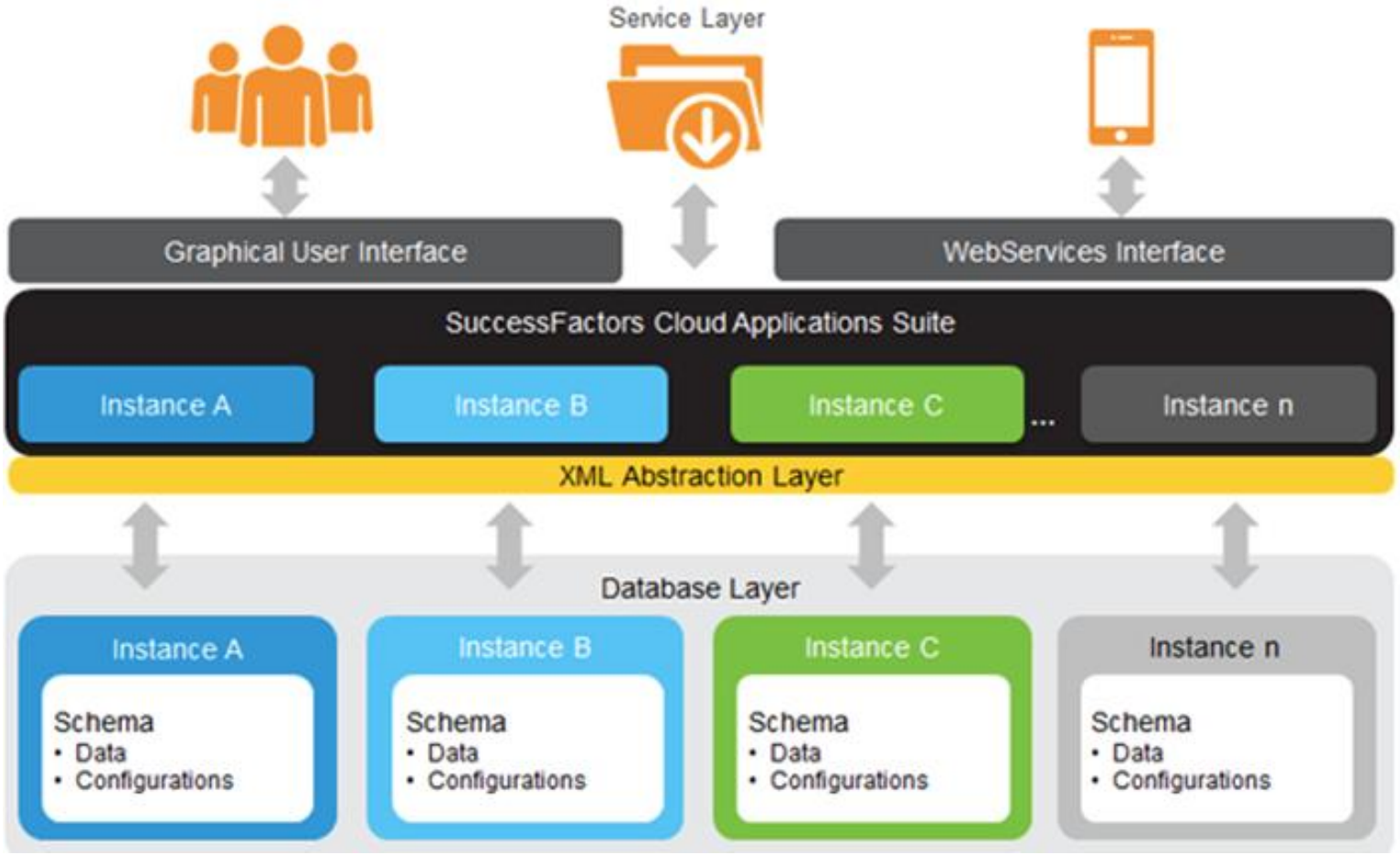
SalesForce Offerings

Seven Major Components

SalesForce Offerings

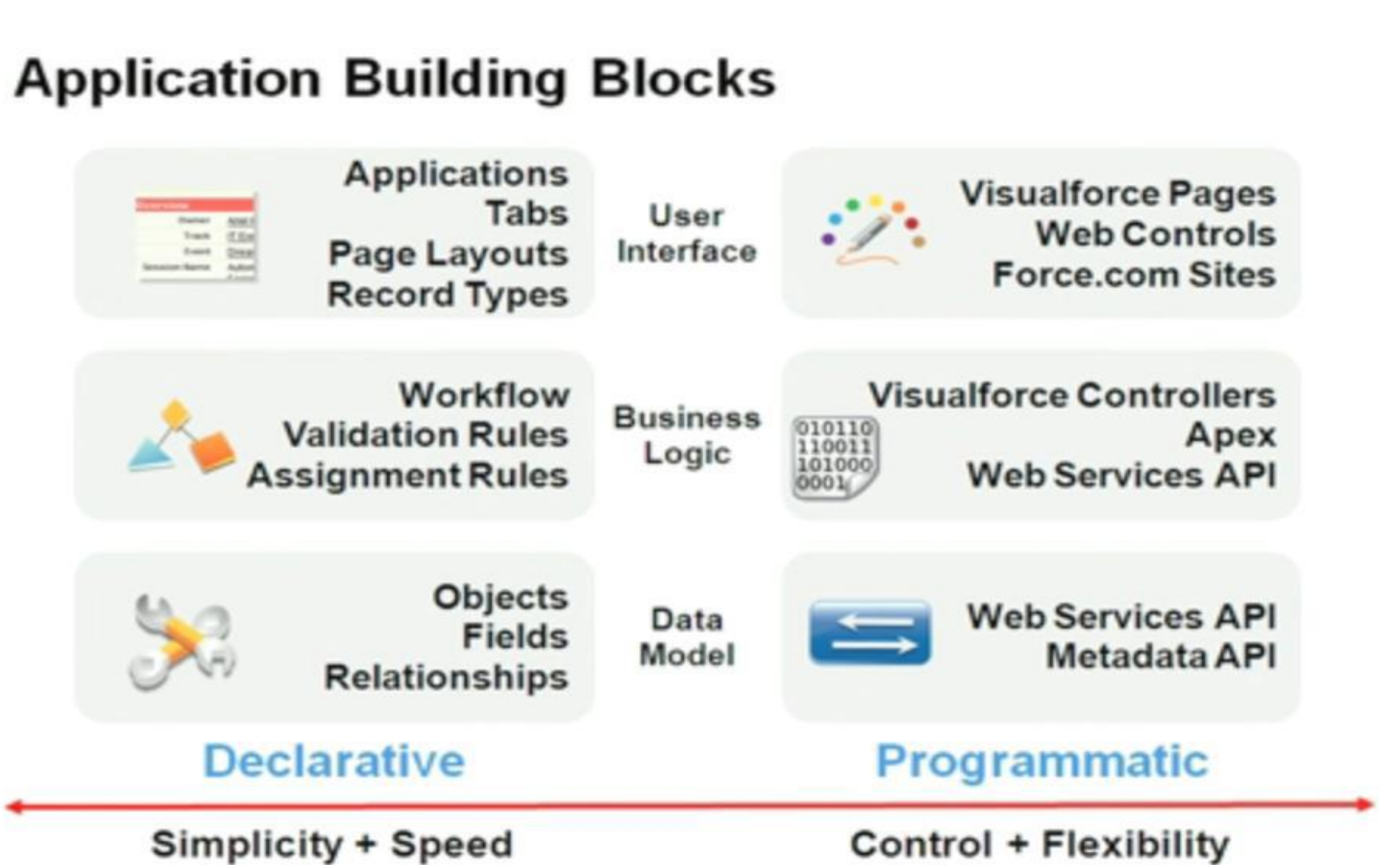
- SalesForce's CRM platform is composed of several services:
  - **Marketing Cloud**
    - Run the marketing efforts & campaigns effectively - get more leads.
  - **Sales Cloud**
    - Sale smarter and faster using the CRM software
  - **Service Cloud**
    - Support all customers efficiently, anytime, anywhere
  - **Community Cloud**
    - Engage employees, partners, and customers under one umbrella
  - **Einstein Analytics**
    - Perform Data Analytics on any data from any device
  - **SalesForce Platform**
    - Build custom CRM applications using the SF Cloud Platform
  - **SalesForce IoT**
    - Support & integrate Internet-of-Things with customer management

Multi-Tenant Architecture



- Personal Credentials or single sign-on
- Distinct application instance per client forcing memory segregation
- Distinct database schema per client → No commingling of data
- Scalable and secure multi-tenant model with high configurability

SalesForce CRM



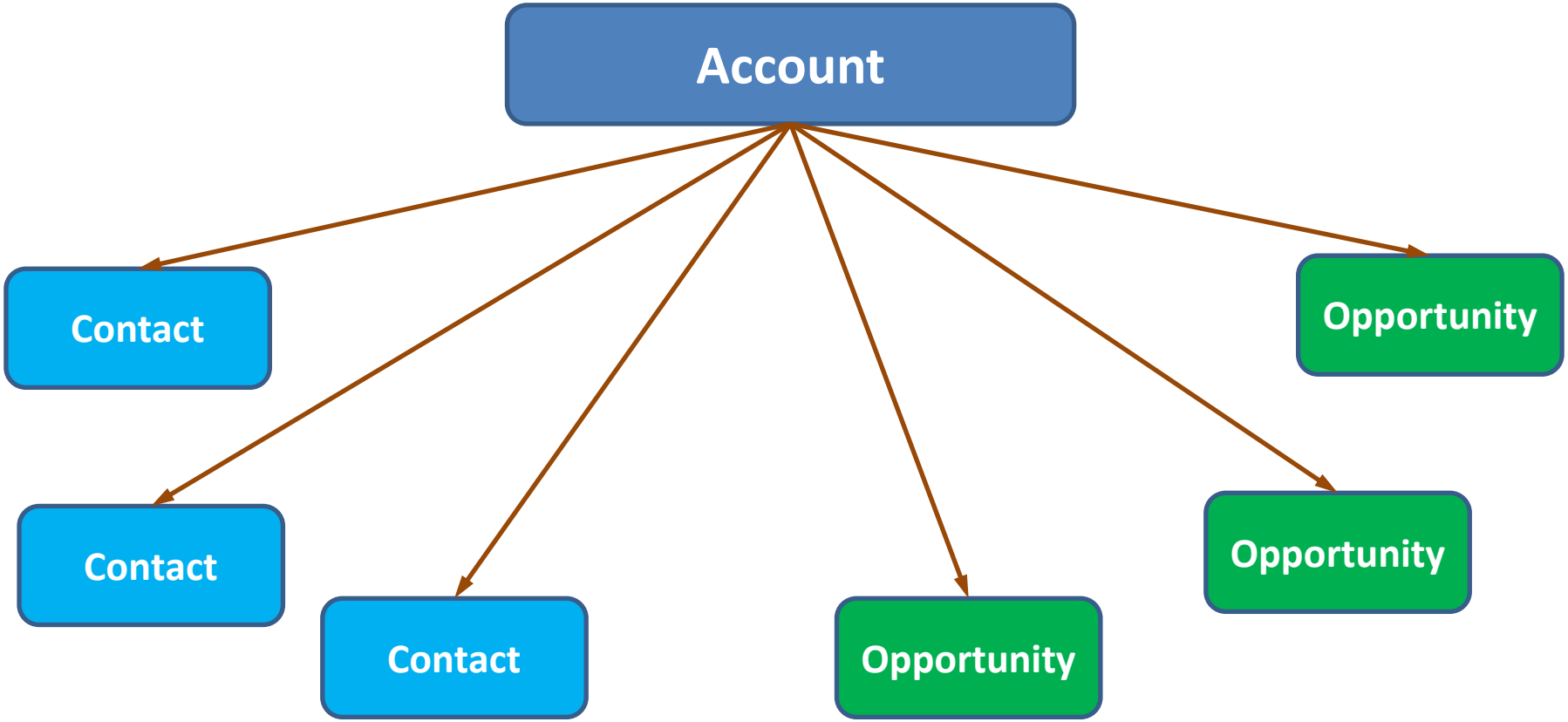
SalesForce Data Model (Objects)

- **Account Object**
  - An Account is any company, NGO, working group, entity, partner, etc., that you already have or hope to have a relationship with.
  - Accounts are *central* in using SalesForce. Contacts and Opportunities are directly linked to Accounts.
- **Contact Object**
  - Any person belonging to any account with whom you already have or hope to have a business relationship.
  - Can include partners, suppliers, vendors, customers, prospects, etc., -- anyone with a *pulse*!
- **Opportunities Object**
  - Any potential or realized business opportunity
  - Roughly similar to an order form, agreement, or invoice
  - Tracks potential, as well as *won* or *lost* business opportunities
  - Can be used by Finance Team to estimate cash flow
- **Activities Object**
  - Used to record interactions with customers or prospects
  - Open Activities, create New Tasks, or schedule future Tasks
  - Activity history: Log details of completed activities, log calls.

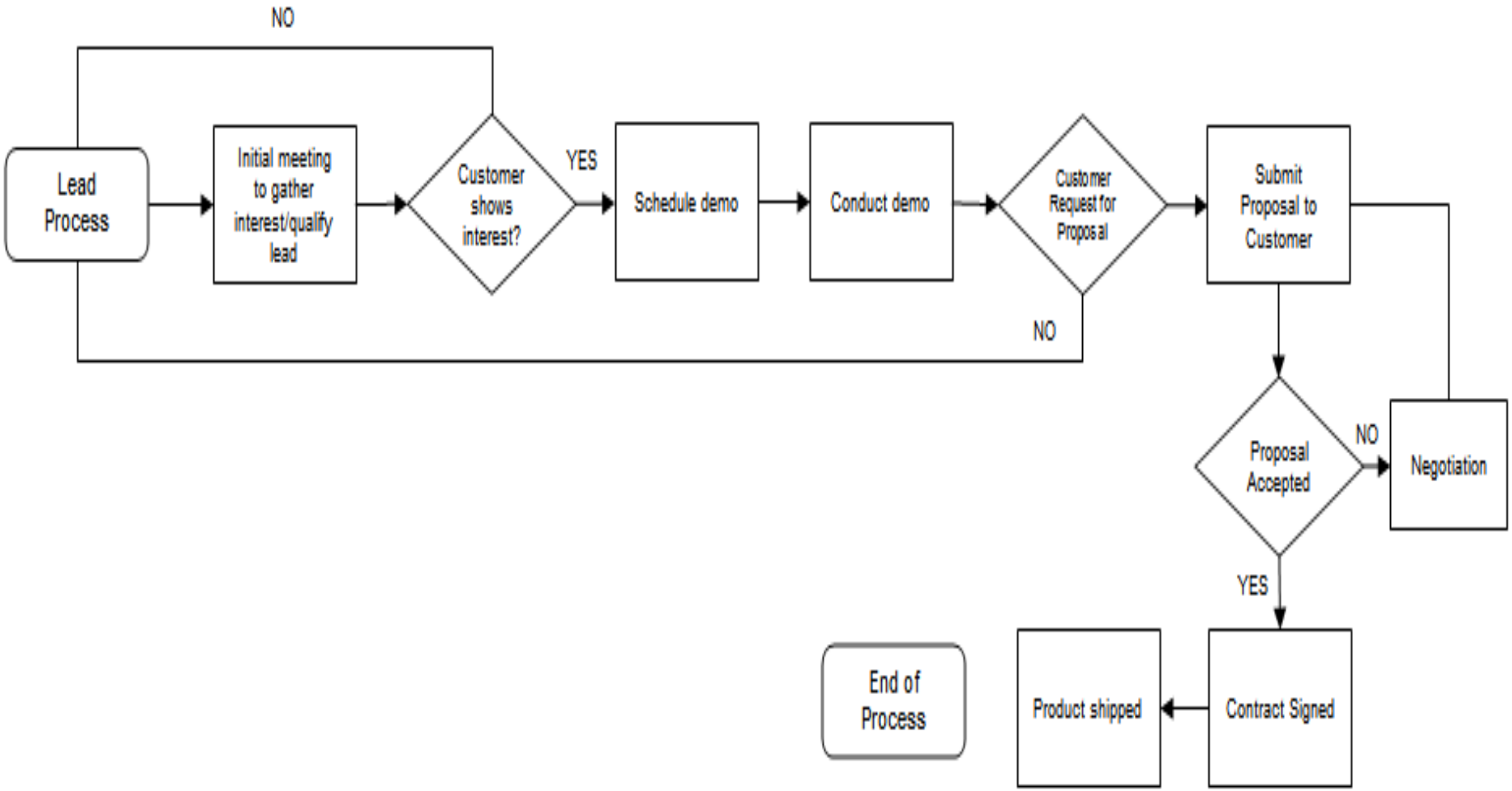
SalesForce Data Model (Objects)

- **Sales Objects:** includes accounts, contacts, opportunities, leads, campaigns, and other related objects
- **Task and Event Objects**: includes tasks and events and their related objects
- **Support Objects**: includes cases and solutions and their related objects
- **Document, Note, and Attachment Objects**: includes documents, notes, and attachments and their related objects
- **User and Profile Objects**: includes users, profiles, and roles
- **Record Type Objects**: includes record types and business processes and their related objects
- **Product and Schedule Objects**: includes opportunities, products, and schedules
- **Sharing and Team Selling Objects**: includes account teams, sales teams, and sharing objects
- **Territory Management**: includes territories and related objects
- **Process Objects**: includes approval processes and related objects
- **Content Objects**: includes content and workspaces and their related objects.
- **Salesforce Chatter Objects**: includes objects related to feeds.

SalesForce Structure



Basic Sale Process



Leads - Most Important in Business!

- Leads are optionally used to represent a *potential* relationship than an *actual* relationship.
- Who is a *Lead* and who is a *Contact* should be defined by the organization and strictly followed by all.
- Someone who is interested in the organization's products / services :- Potential buyers, donors, grantees, volunteers … **...**
- Leads can be generated from a variety of sources
  - Ad Campaigns leading to Web-to-Lead Form
  - List of leads purchased from 3rd party sources
- Leads who develop a relationship are converted to Contacts / Opportunities, and optionally to Accounts and Relationships
- Organization should define when to convert a Lead

Marketing via SalesForce

Send email blasts

Google AdWords

Online Advertising

Include URL directing leads to online form

Online form collects data and tracks in Salesforce

Create an automated email notification which s immediately sent back to end user

Create an assignment rule that delegates a task & email notification to a co-worker to contact lead within minutes

SalesForce Mass Email

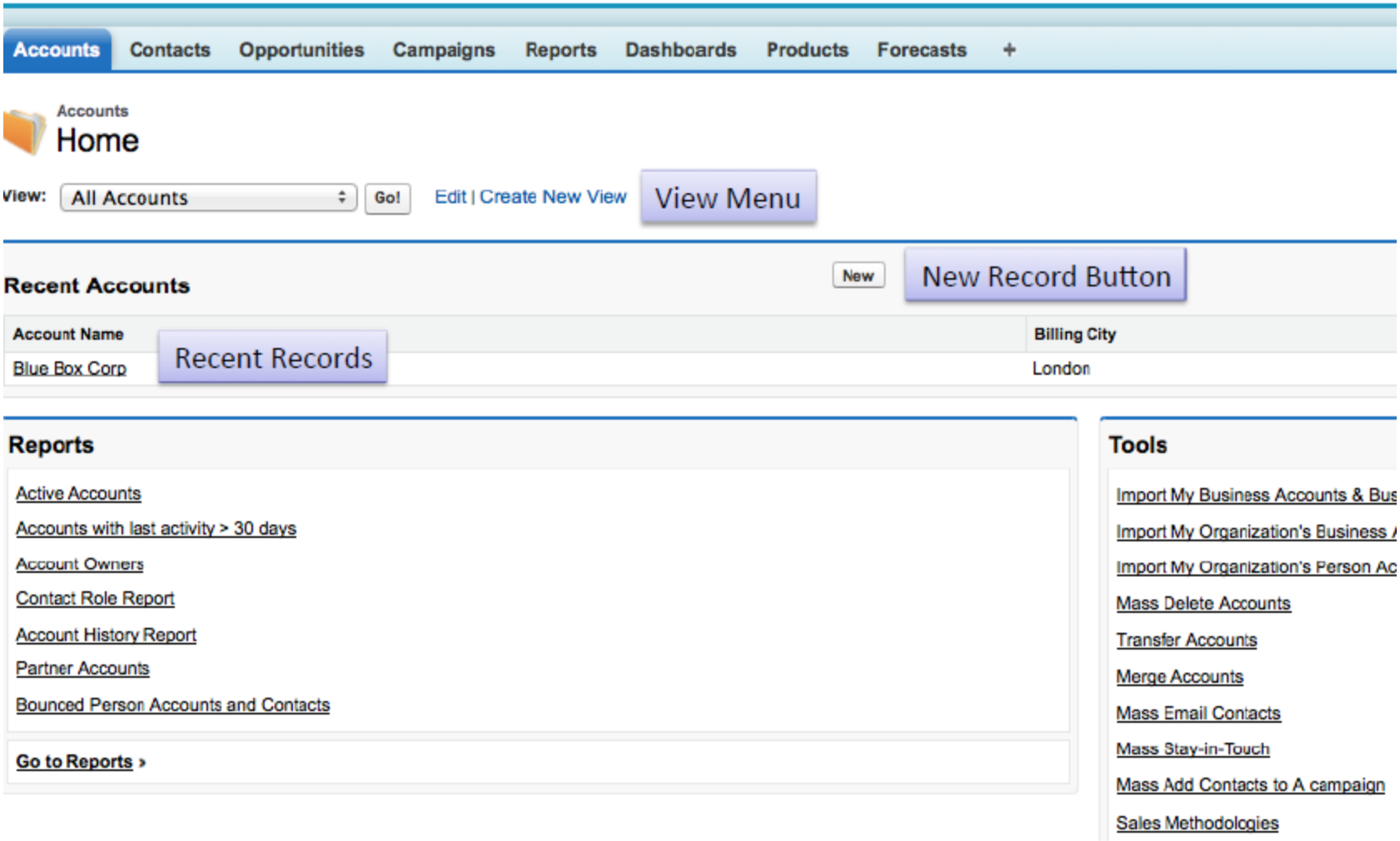
The salesforce mass email and web-to-lead forms, enables:

SalesForce Web-to-Lead

- Easily create web inquiry form for potential customers to post a request on your website
- Use Auto Response Rules
  - Create a message to be sent to the lead upon submission of the inquiry form on website
- Use Assignment & Workflow Rules
  - Specify criteria to assign tasks to specific co-workers
  - Increase the rate of conversion by using workflow rules to respond back to the lead within minutes

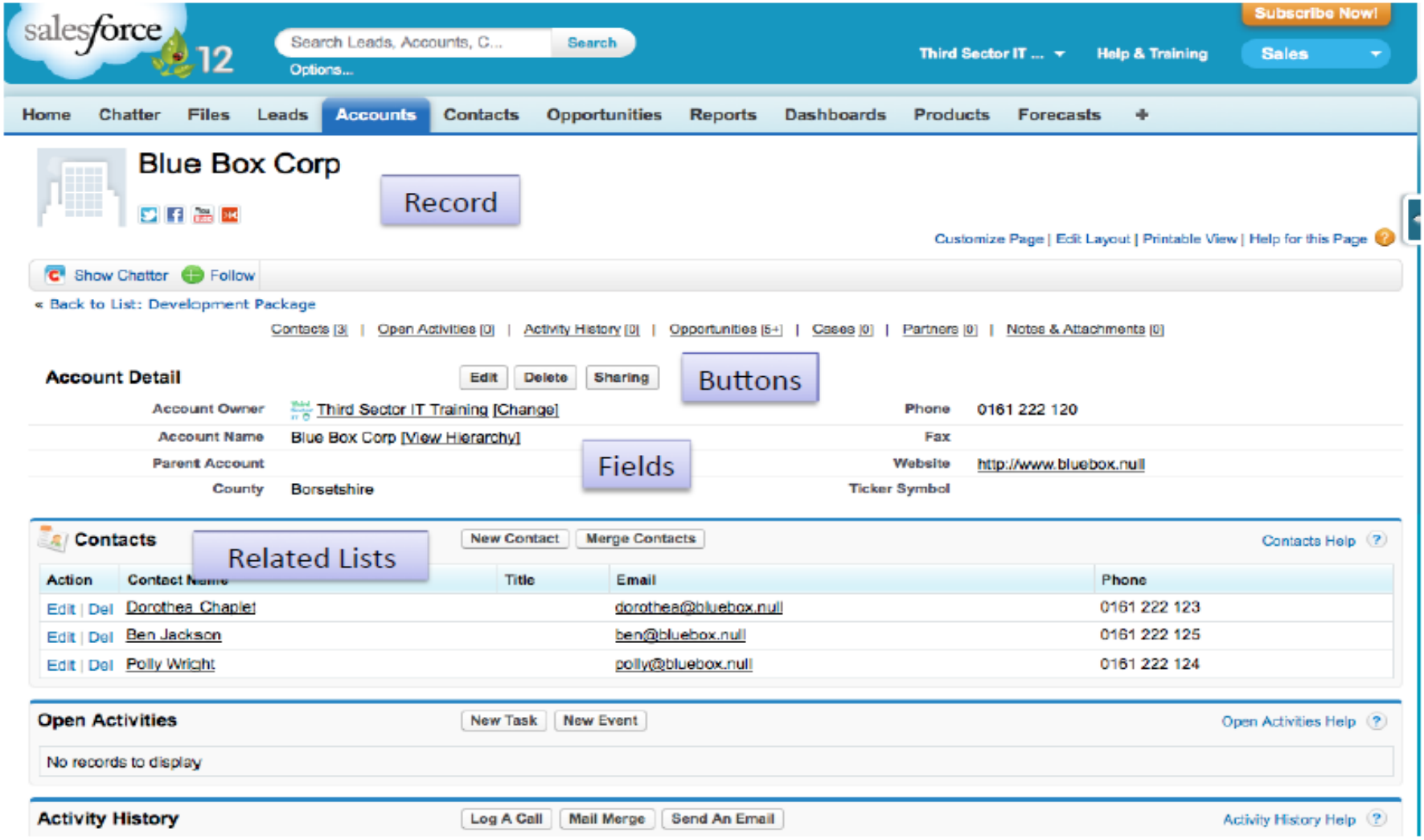
SalesForce Interface

Tab Home Page

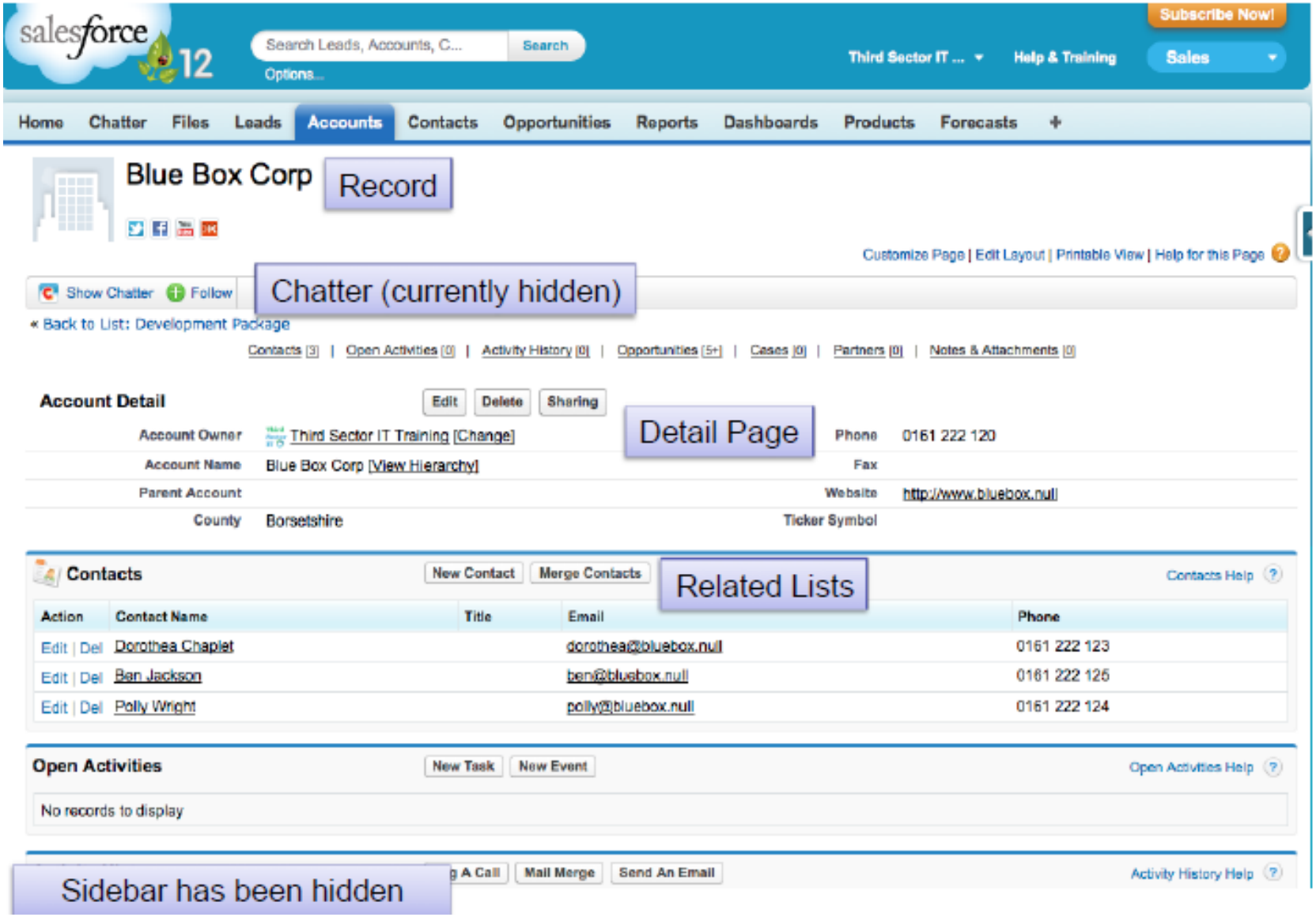


SalesForce Interface

Record Detail Page



SalesForce Interface



Force.com

Force.com

- Force.com is a cloud computing PaaS from SalesForce that developers use to build multitenant applications hosted on their servers as a service.
- It is targeted towards corporate application developers and independent software vendors (ISVs).
- Unlike the other PaaS offerings, it does not expose developers directly to its own infrastructure.
- Developers do not provision CPU time, disk, or instances of running operating systems.
- Instead, Force.com provides a custom application platform centered around the relational database.

Force.com

- Force.com is a part of SalesForce.com which is established as a SaaS based CRM vendor
- However, Force.com is unrelated specifically to CRM. It provides the infrastructure commonly needed for any business application.
- It allows developing customized solutions for the unique requirements of a business through a combination of code and configuration.
- Force.com provides all necessary plumbing for security, user identity, logging, profiling, integration, data storage, transactions, workflow, and reporting.

Force.com : 9 Key Technologies

- **Multitenant Architecture**:
  - An application model in which all users and apps share a single, common infrastructure and code base.
- **Metadata-driven Development Model**:
  - A development model that allows applications to be defined as declarative “blueprints,” with no (or much less) code required.
  - Data models, objects, forms, workflows (and much more) etc., are defined by metadata.
- **API Access**:
  - Several APIs (e.g., RESTful Bulk API, Data Loader API, Metadata API, Chatter REST API) provide direct access to all data stored in Lightning Platform from virtually any programming language and platform.

Force.com : 9 Key Technologies

- **Apex**:
  - The world's first on-demand programming language, which runs in the cloud on the Lightning Platform servers.
- **Visualforce**:
  - A framework for creating feature-rich user interfaces for apps in the cloud.
- **Mobile Access**:
  - With SalesForce mobile apps, you can access custom apps built using the Lightning platform's GUI driven development tools.
  - Users can access those apps on their mobile devices—and you don't have to learn any mobile programming languages.

Force.com : 9 Key Technologies

- **AppExchange Directory**:
  - A Web directory where hundreds of Lightning Platform apps are available to Salesforce customers to review, demo, comment upon, and/or install.
  - Developers can submit their apps for listing on the directory if they want to share them with the community.
- **SalesForce Object Query Language (SOQL)**
  - An SQL-like language for querying the database
- **SalesForce Object Search Language (SOSL)**
  - Enables developers to perform full-text search on descriptive data (some SQL implementations support full-text search)

Multitenant Architecture of SalesForce

- Multitenancy is a software architecture where a single instance of the software runs on a server, serving multiple client-organizations (tenants) from the same code and database.
  - **Shared Infrastructure:** Every customer (or tenant) of Force.com shares the same infrastructure.
  - **Single version:** only one version of the Force.com platform is used to deliver applications of all sizes and shapes.
  - **Continuous, Zero-cost Improvements:** When Force.com is upgraded to include new features or bug fixes, the upgrade is automatically enabled in every customer's logical environment with zero to minimal effort required.

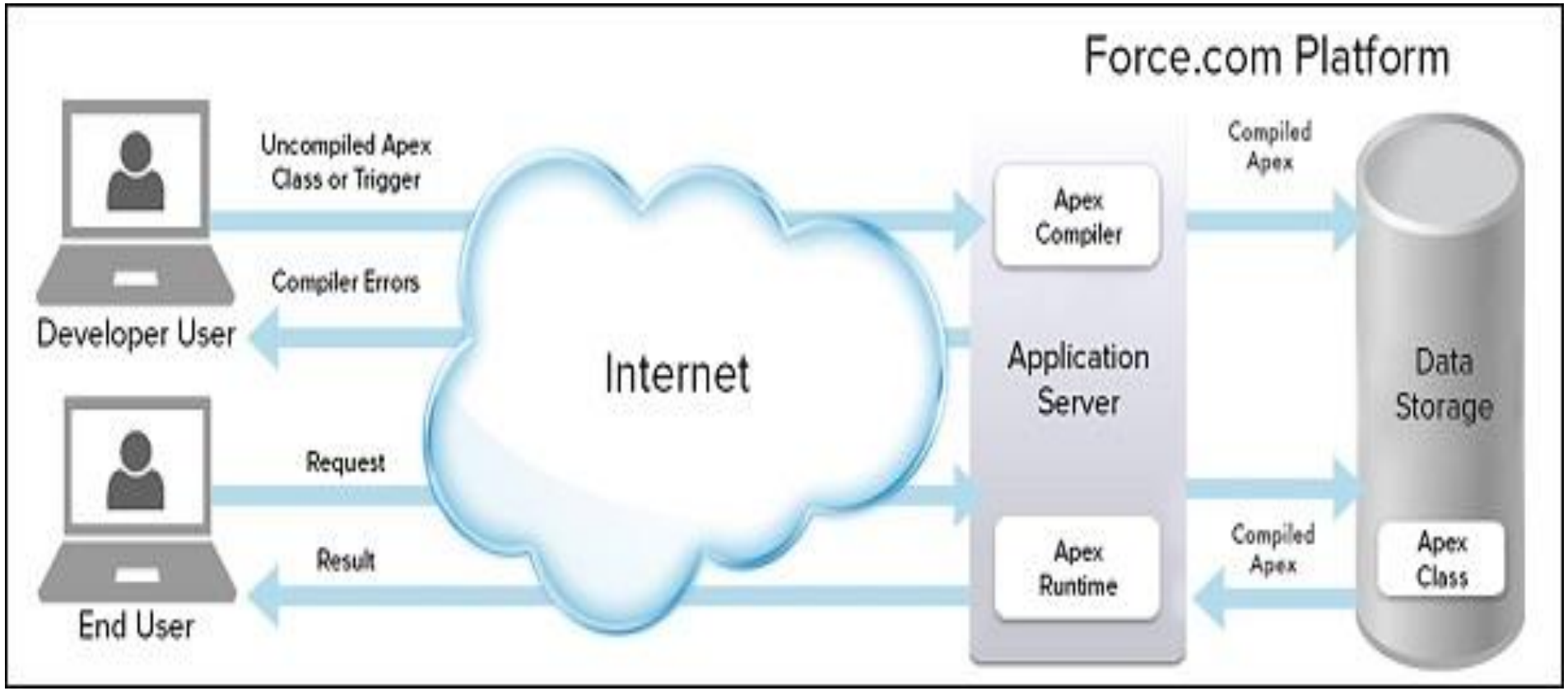
VisualForce

- Visualforce is a MVC framework that allows developers to build sophisticated, custom user interfaces that can be hosted natively on the Force.com platform.
- The Visualforce framework includes a tag-based markup language, similar to HTML that is easy to learn.
- It also includes a set of server-side “standard controllers” that make database operations very simple to perform.
- MVC - Model-View-Controller Architecture
  - Model: Salesforce Objects, Metadata
  - View: Tabs, Forms and Visualforce pages
  - Controller : Workflows, Apex Controllers, Triggers

Apex

- Apex is a strongly typed, object-oriented programming language that has a Java-like syntax and acts like database stored procedures.
- Allows developers to execute flow and transaction control statements on the Force.com platform server along with calls to the Force.com API.
- Apex runs in a multitenant environment and enables developers to add business logic as per the needs.
- Apex is upgraded automatically with each SalesForce release - no manual upgrade needed.
  - Check the following for an overview on Apex:
  - https://www.tutorialspoint.com/apex/apex_overview.htm

Apex Execution Model



Microsoft Office Live

Currently known as Office 365

Microsoft Office 365 (SaaS)

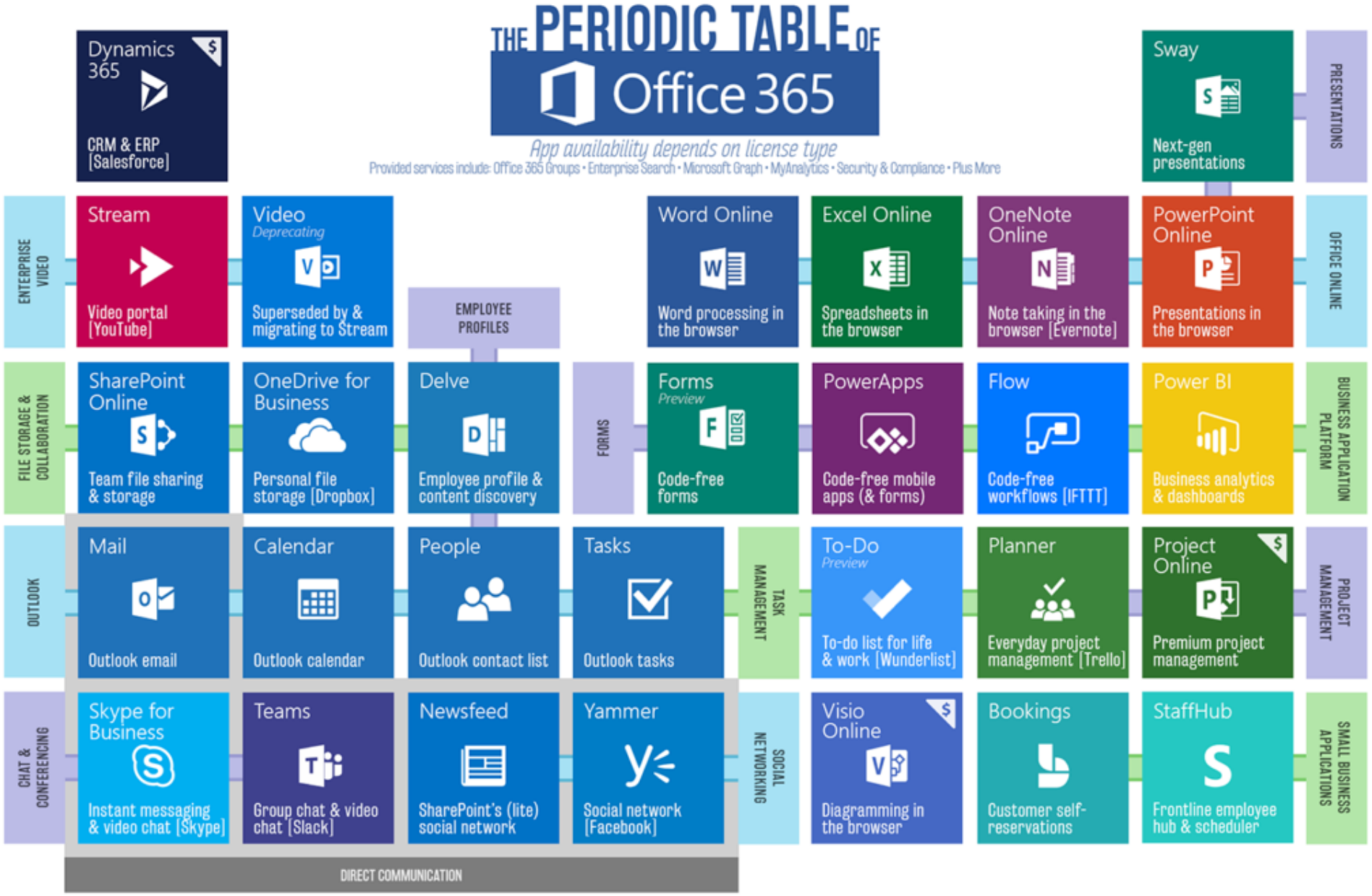
- The MS Office 365 suite is a cloud based online version of the traditional MS Office software (SaaS Model).
- It is subscription-based and includes MS Office, MS Exchange, SharePoint, Lync and MS Office Web Apps.
- Administrators access the MS Office 365 suite from a web-based portal to create new user accounts, set up permissions, and control access to features.
- Users can use the software packages included in MS Office 365 suite through a web browser.
- Access anywhere, anytime – needs Internet connection.

Common Office 365 Features

Office 365 features vary widely depending on the subscription plan. Below are some of the common features:

- **Office Suite** (Word, Excel, PowerPoint, Outlook, OneNote, Publisher, Skype for Business, Access)
- **Exchange Online** (email, calendar, tasks)
- **SharePoint Online** (web portal for collaboration)
- **Yammer** (enterprise social networking)
- **OneDrive for Business** (cloud file storage)
- **Planner** (project management)
- **Power BI** (business intelligence)
- **Delve** (social document discovery)
- **Video** (a private video library)
- **Sway** (a tool for creating reports, presentations and newsletters)

Common Office 365 Features



Office 365 – Plans & Pricing

- Office 365 Business:
  - Rs. 545 per month (annual commitment)
  - Outlook, Word, Excel, PowerPoint, OneNote, Access, OneDrive
- Office 365 Business Premium
  - Rs. 660 per month (annual commitment)
  - Outlook, Word, Excel, PowerPoint, OneNote, Access, OneDrive, Exchange, SharePoint, Skype for Business, MS Teams
- Office 365 Business Essentials
  - Rs. 125 per month (annual commitment)
  - MS Office Applications – NOT INCLUDED
  - Includes OneDrive, Exchange, SharePoint, Skype for Business, MS Teams

Office 365 – Pros & Cons

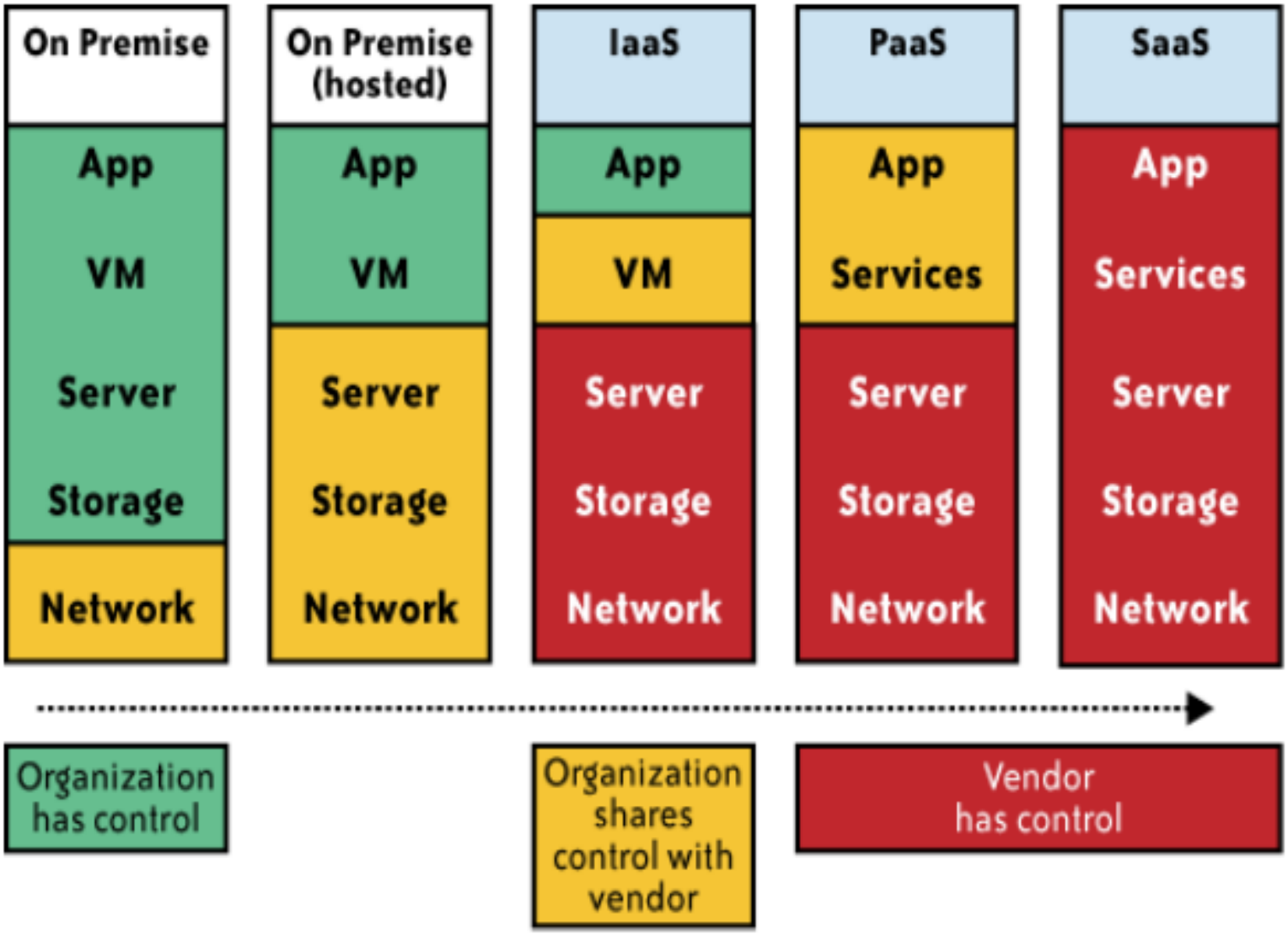
- Advantages:
  - Low Cost, Subscription based, No Installation Required
  - Work From Anywhere, Anytime
  - Seamless Collaboration to Maintain Productivity
  - Robust Security Features for Extra Protection
  - No worries for Backups of data
  - Software & Program Familiarity
  - Multiple Plans Tailored to Businesses
  - Automatic availability of upgrades
- Disadvantages:
  - Internet Issues Can Disrupt Productivity
  - Subscription based → Renewals need to be remembered
  - Data Privacy & Control → Not In Your Hands
  - Compatibility Issues

Windows Live Mesh

- Formerly knows as Windows Live FolderShare, LiveSync…
- Was part of the Windows Live Essentials 2011 suite of software.
- Discontinued since September 2013
- Allowed files and folders between two or more computers to be in sync with each other via SkyDrive (*now called OneDrive*).
- Live Mesh also enabled remote desktop access via the Internet.
- Features:
  - Ability to sync up to 200 folders with 100,000 files each (each file up to 40 GB) for PC-to-PC synchronization
  - Ability to sync up to 5 GB of files to "SkyDrive synced storage" in the cloud
  - Remote Desktop access via Windows LiveMesh
  - PC-to-PC synchronisation of application settings for applications such as:
    - Windows Internet Explorer
    - Microsoft Office - synchronisation of dictionaries, Outlook email signatures, styles and templates between computers

Cloud Security

Governance in the Cloud



Cloud Security

- The cloud acts as a big black box, nothing inside the cloud is visible to the clients.
- Clients have no idea or control over what happens inside a cloud → loss of control.
- Clients have to rely on a 3rd party for security of their applications, processes, and data.
- Clouds are still subject to traditional security issues:
  - Confidentiality, Integrity, Availability, and Privacy
  - Additional threats/attacks due to the nature of cloud
- Cloud provider may be honest, but there may be malicious employees who can tamper with the VMs and violate confidentiality and integrity.

Cloud Security

- Why "Cloud" is still seen with doubt by clients?
  - Main concern: SECURITY
- Most security problems in the cloud stem from:
  - Loss of Control (client has diminishing control)
  - Lack of Trust (policies & mechanisms)
  - Multi-Tenancy (no idea about integrity of the co-tenants)
- It seems that these problems exist mainly in cloud deployment models managed by a 3rd party.
- However, self-managed clouds (private clouds) may still have a number of security & privacy issues.

The 5 Security Issues in the Cloud

1. Consumer's loss of control
   - Data, applications, resources are located with provider
   - User identity management is handled by the cloud
   - Users' access-control rules, security policies and enforcement are also managed by the cloud provider
2. Consumer has to rely on the CSP to ensure
   - Data security and privacy
   - Resource availability
   - Monitoring and repairing of services/resources
3. Conflict between tenants' opposing goals
   - Tenants share a pool of resources may have opposing goals
   - An attacker can legitimately be in the same physical machine as the target.

The 5 Security Issues in the Cloud

4. Privacy issues raised via massive data mining
   - Cloud now stores data from a lot of clients, and can run data mining algorithms to get large amounts of critical information on clients with malicious intent.
5. Increased attack surface
   - Entity outside the organization stores data and performs all computation on the data in the cloud
   - Attackers can target the communication link between cloud provider and client
   - Cloud administrative accounts may be hacked
   - Cloud provider employees can be phished

Basic Components of Security

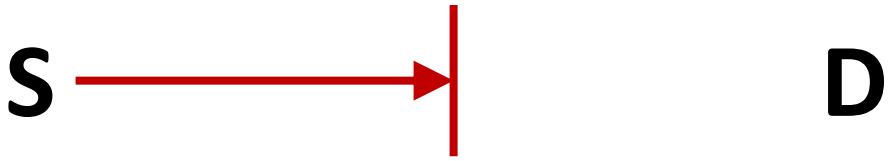
- **Availability**
  - Enabling access to resources to authorized users
  - Permanence, non-erasure, no loss of data
- **Confidentiality**
  - Keeping data hidden from unauthorized users
- **Integrity**
  - Data Integrity (data is correct as it is supposed to be)
- **Authentication**
  - Data is from authenticated source (trusted origin)
  - Data is used only by authenticated (trusted) users

Security Attacks

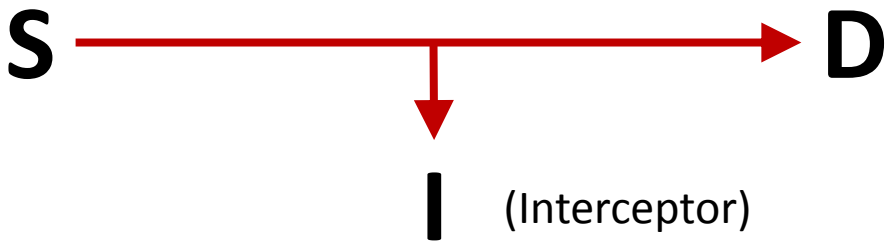
- Any *action* that compromises with the safety and security of information is called an *attack*.
- Attacks are broadly of 4 types
  - Interruption
  - Interception
  - Modification
  - Fabrication
- Basic Model:

Security Attacks

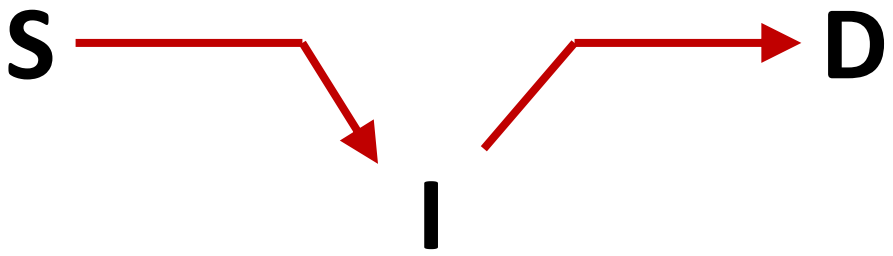
- **Interruption**:
  - Attack on availability



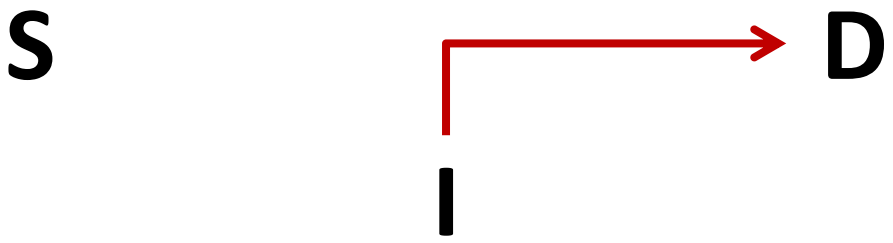
- **Interception**:
  - Attack on confidentiality



- **Modification**
  - Attack on integrity



- **Fabrication**
  - Attack on authenticity



Classes of Threats

- **Disclosure**:
  - Confidential information getting into hands of others
  - Snooping
- **Deception**:
  - Modification, Spoofing, Repudiation of Origin
  - Denial of Service
- **Disruption:**
  - Denial of Service
  - Modification
- **Usurpation**
  - Taking control of something without having right on the same
  - Modification, Spoofing, Delay, Denial of Service

Security Policies & Mechanisms

- Policy says what is *allowed* and what is *not allowed*.
- Policies define security aspects for
  - Applications, Processes, Data, Users, Network, … …
- Mechanisms enforce security policies.
- Composition of Policies must be proper.
  - Discrepancies/conflicts in policies → security vulnerabilities

Difference between Threat and Attack:

- Threat (Vulnerability)
  - Some kind of breach that "may" be possible
- Attack
  - Actually a breach happens (by exploiting a vulnerability)

Goals of Security

- **Prevention**
  - Preventing attackers from violating security policies
  - Preventing attackers from bypassing security mechanisms
- **Detection**
  - Recognize an event of violation of security policy or mechanism
- **Recovery**
  - Stop the attack (first priority)
  - Assess and repair the damage caused by the attack
  - Continue to function normally after the attack
- **Protection**
  - Modify policies to protect from further attacks of same type
  - Develop & enforce mechanisms to protect the systems

Roles of Security

The security infrastructure of a CSP must fulfill:

- **Confidentiality**
  - Protection against loss of privacy (leakage of sensitive data)
- **Integrity**
  - Protection against data corruption / alteration
- **Availability**
  - Protection against denial of service
- **Authentication**
  - Whether an action by a user should be allowed or not
- **Non-repudiation**
  - Ability to trace back what has happened & prevent denial of service
- **Safety**
  - Protection against tampering, damage, and theft of data

Security in the Cloud

- **Infrastructure Security**
  - Network Level
  - Host Level
  - Application Level
- **Data Security**
  - Transmission, Storage & Security of Customer Data
  - Security of Service Provider Data
- **Identity & Access Management (IAM)**
  - IAM Challenges
  - IAM Architecture & Practice
- **Privacy**
- **Audit and Compliance**

Infrastructure Security : Network Level

- For private clouds, the network topology and existing security measures may not change significantly.
- However, if public cloud services are used, the security requirements of the organization will change.
- This will require changes to existing network topology, configuration, security mechanisms, firewalls, routing …
- Four significant risk factors arise:
  - Ensuring confidentiality and integrity of data-in-transit
  - Ensuring proper access control to resources on public cloud
  - Ensuring availability of the Internet-facing resources
  - Replacing the existing network zones and tiers with domains

Infrastructure Security : Network Level

- Ensuring confidentiality and integrity of data-in-transit:
  - In the Intranet, HTTP queries may be fine, however, transactions going out of the intranet, must use HTTPS
    - May require changes to applications, settings, firewall configurations
  - Detailed logging of network activities may not be available
    - Organization's N/w administrators generally do not have access to detailed network activity logs inside the CSP's network infrastructure
- Ensuring proper access control to resources:
  - IP addresses are "aged" and shared between tenants (Elastic IP)
  - Requires DNS changes, flushing of DNS cache files, and routing reconfiguration → involves a lag time (downtime?)
  - Private (non-routable) IP addresses may be used by other tenants to reach your resources

Infrastructure Security : Network Level

- Ensuring availability of the Internet-facing resources:
  - BGP (Border Gateway Protocol) Prefix Hijacking
    - Falsification of Network Layer Reachability Information
    - Involves announcement of address space that belongs to someone else
    - Usually due to misconfiguration of routing (100s occur every month)
  - Deliberate DNS attacks, DDoS Attacks (external/internal)
- Replacing the existing network zones and tiers with domains (for access over HTTP/HTTPS):
  - Traditional isolation of network *zones* and *tiers* are no longer applicable in PaaS and IaaS models.
  - Public cloud follows *security groups* and *security domains*
  - Most cloud providers follow *security domain* model

Infrastructure Security : Host Level

- In SaaS and PaaS models, the CSP is responsible for host level security, but in IaaS, customer is responsible.
- Several security threats possible:
  - **VM Escape** (isolation of VMs compromised)
  - **Configuration drift**
    - Primary H/w and S/w configurations become different in some way due to a recovery or secondary configuration or vice versa
  - **Insider threats** due to weak access control to hypervisor
  - **Security of Virtualization Software**???
    - User of one VM may get into your VM and execute commands
  - **Security of Customer Guest OS**???
    - User of the Guest OS may get into other VMs and execute commands
  - **Security of the VM Management APIs**???
    - Attacker can exploit bug in VM Management APIs to destroy your VM
    - Stealing SSH Keys, Hijacking of Accounts, Deploying Trojans … …

Infrastructure Security : Application Level

- Developing applications for deployment in the cloud require re-evaluation of existing security practices and standards established by an organization.
- Web applications built and deployed in public cloud platforms are subjected to higher security risks.
- Various application level security threats possible:
  - Cross-Site-Scripting (XSS) Attacks
  - SQL Injection Attack
  - Malicious File Inclusion
  - DoS, DDoS, EDoS, and more… …
- Most application level vulnerabilities result from design flaws or programming errors.

Infrastructure Security : Application Level

- End-User Security
  - Generally, the customer is responsible for end-user security
  - Have your antivirus, antimalware, firewalls, IDS/IPS etc.
- Responsibility for Web Application Security on the Cloud
  - Depends on service delivery model and SLA
- In SaaS model, the CSP is fully responsible
  - but up to the extent mentioned in the SLA
  - SLA is an important contract between customer and CSP
- In PaaS model, both customer and CSP are responsible
  - Security of the PaaS platform itself - by the CSP
  - Security of the applications deployed on PaaS - by Customer

Data Security

- Data Security is one of the main concern for any business to move into cloud services.
- Security of data stored in cloud is much more important when using cloud services at all levels (IaaS, PaaS, SaaS)
- Six major aspects of data security:
  1. Data-in-Transit
  2. Data-at-Rest
  3. Data Processing
  4. Data Lineage
  5. Data Provenance
  6. Data Remanence

Data Security

- **Data-in-Transit**:
  - All forms of attacks can happen on data-in-transit in between the user and the cloud service.
  - Any data communication without using an established encryption algorithm poses a primary security risk.
  - Still, just encrypting the data-in-transit may not be sufficient!
  - Encrypting data but using a *non-secured* protocol may provide confidentiality, but does not guarantee integrity.
  - Secured protocols must be used for all communication
    - HTTPS over HTTP
    - FTPS over FTP
    - Secure Copy (SCP)

Data Security

- **Data-at-Rest**:
  - Data of cloud applications recorded in the storage media is referred to data-at-rest, the volume is generally high.
  - Encrypting all the data-at-rest is the obvious choice for administrators, but it may not as simple as it sounds!
  - In IaaS or cloud services used for simple storage, encrypting data-at-rest may be fine for most purposes.
  - However, data-at-rest used by a cloud-based application is generally not encrypted to enable indexing or searching.
  - Data-at-rest in unencrypted form poses a big security risk.
  - Commingling of unencrypted data (e.g. Google BigTable) poses a bigger risk due to multitenancy in a public cloud scenario.
  - Application vulnerabilities can expose unencrypted data.

Data Security

- **Data Processing**:
  - Storing data in encrypted form in the cloud is required.
  - However, for any processing on the cloud, the data must be first unencrypted to be used by the applications.
  - The duration for which data remains in unencrypted form (whether in memory or in storage) poses a security risk.
  - It must be ensured that data be unencrypted for least amount of time in its life cycle, even with proper isolation of tenants.
  - Repeated encryption/unencryption of frequently used (sensitive) data may slow down the applications considerably.
  - Homomorphic encryption algorithms have been developed which allows processing data without unencryption.
  - However, these algorithms also require immense computational effort.

Data Security

- **Data Lineage**:
  - *Data Lineage* refers to following the path of data, i.e., accurate mapping of application data flows or data path visualization.
  - Important for audit and compliance purposes, whether internal, external, or regulatory.
  - Providing exact data-lineage can be a very time consuming process even when the cloud is under organization's control.
  - Attempting to provide accurate reporting on data lineage for a public cloud service is really not possible.
  - Auditors have little means to figure out:
    - Where was the data physically located during its entire path?
    - What was the state of the systems through which data flowed?
    - What other applications or tenants were there during those times?

Data Security

- **Data Provenance**:
  - Integrity and Provenance are two different aspects
  - *Integrity* means data has not been changed in unauthorized manner by an unauthorized person.
  - *Provenance* means data not only has integrity, but also is accurately computed by the correct application/component.
  - Proving data provenance is an essential requirement in financial and scientific applications.
  - The typical multi-tenant shared environment of cloud makes it impossible to prove data provenance.
  - Ability to track the systems used or their state at the times the were used for a computation is impossible to trace back.
  - Only IP address and general location is grossly insufficient.

Data Security

- **Data Remanence**:
  - Defined as the residual representation of data that has been nominally erased or removed in some way.
  - This residue may be due to data being left intact by a nominal delete operation, or physical properties of the storage medium.
  - Can cause inadvertent disclosure of sensitive information, if the storage media is released into an uncontrolled environment (e.g. thrown in trash or resold to another 3rd party)
  - The risk is there in all cloud service models (IaaS/PaaS/SaaS)
  - Even if it poses a serious risk of inadvertent disclosure, most CSPs do not pay the attention *data remanence* requires.
  - Data Remanence is usually not covered within the SLA, or the coverage is fuzzy to provide a zero remanence guarantee.

Data Security : Provider's Data

- A cloud service provider has to collect a lot of private information and meta data of the customers
- Apart from usage, billing, accounting information, lot of personally identifiable data is also collected by CSP.
- CSPs also collect and store huge amount of security related data (non-customer data)
  - System logs, Firewall logs, N/w monitoring logs …
  - File system and database event logs, audit logs …
  - IDS/IPS data, SIEM data
- Customers should be concerned about
  - What data the CSP collects & how they protect that data?
  - As a customer, what access do you have to the data?

Identity & Access Management (IAM)

- Deals with development, practices, and support for
  - Authentication, Authorization, Auditing (AAA)
- In an intranet set up, the trust boundary of organization is static and well defined →easier to control & manage.
- In a cloud environment, the trust boundary becomes dynamic, requiring higher level software controls for:
  - Strong authentication
  - Role/claim based authorization
  - Trusted sources with accurate attributes
  - Identity Federation and Single Sign On (SSO)
  - User activity monitoring & auditing

Identity & Access Management (IAM)

- Why IAM?
  - Improve operational efficiency by helping to automate user on-boarding and repetitive tasks (e.g. password reset)
  - Regulatory and compliance management to to comply with various regulatory, privacy, and data protection requirements (e.g. HIPAA, SOX, ISO-27002 etc.)
- Federated identity is a key IAM component
  - Enables the linking and portability of identity information across trust boundaries.
  - Enables enterprises and cloud service providers to bridge security domains through web single sign-on and federated user provisioning.

Identity & Access Management (IAM)

- IAM Challenges:
  - Managing access for diverse user population (employees, contractors, partners etc.) requiring access to both internally and externally (on cloud) hosted resources.
  - User turnovers: seasonal staff fluctuations, mergers, acquisitions, and business process outsourcing.
  - Existing access policies are seldom defined centrally and consistently applied → disparate directories, complex web of user identities, access rights, and procedures.
  - Architecting a centralized and automated IAM system to handle all possible situations can be quite complex.
  - Building such automated IAM systems is time consuming and require lot of upfront investment.

IAM Architecture & Practices

- IAM is not a monolithic solution that can be easily deployed, rather an aspect of software architecture.
- It can be a large collection of technology components, processes, standards & practices at the enterprise level.

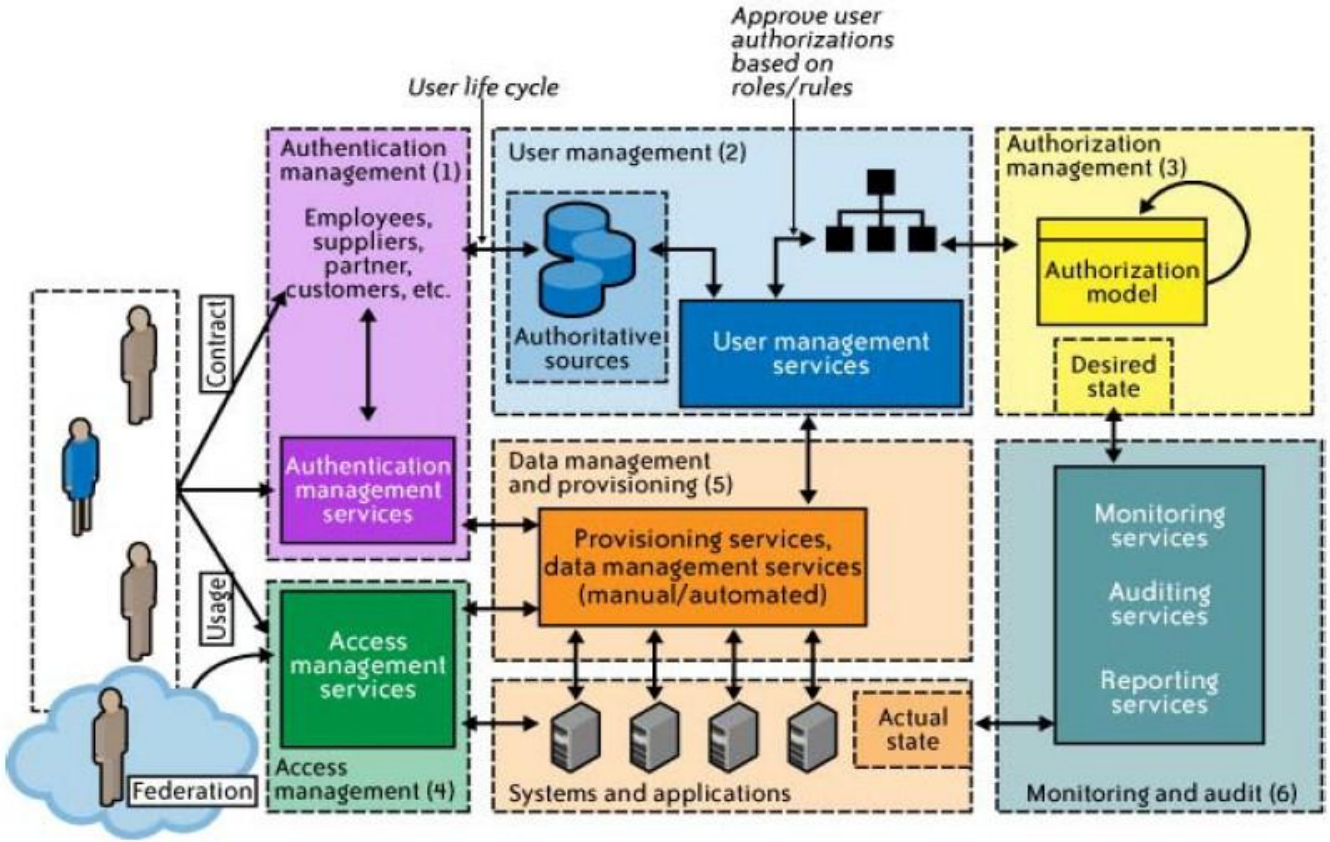
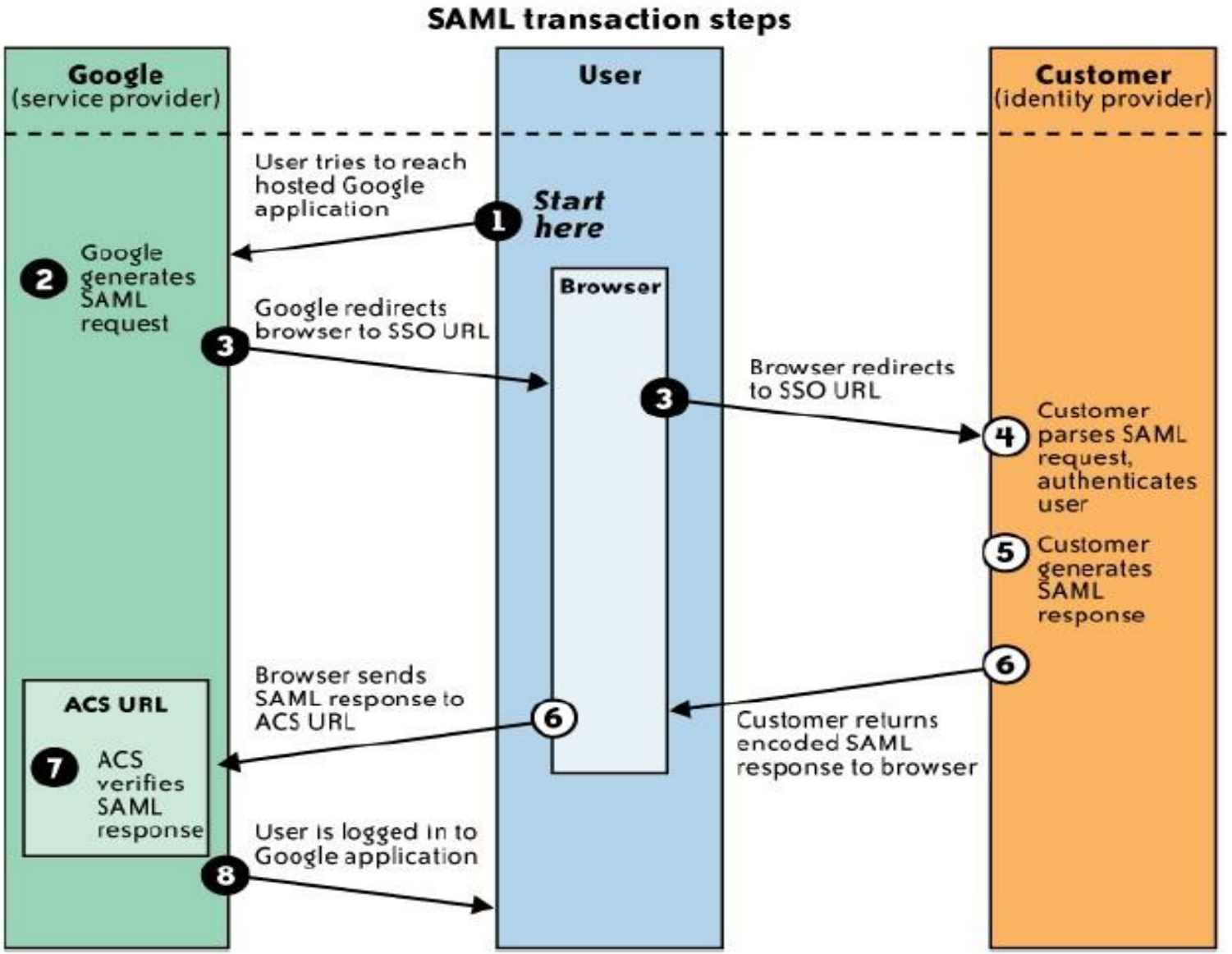


Figure 5.1, page 78
Cloud Security & Privacy
Mather & Kumaraswamy

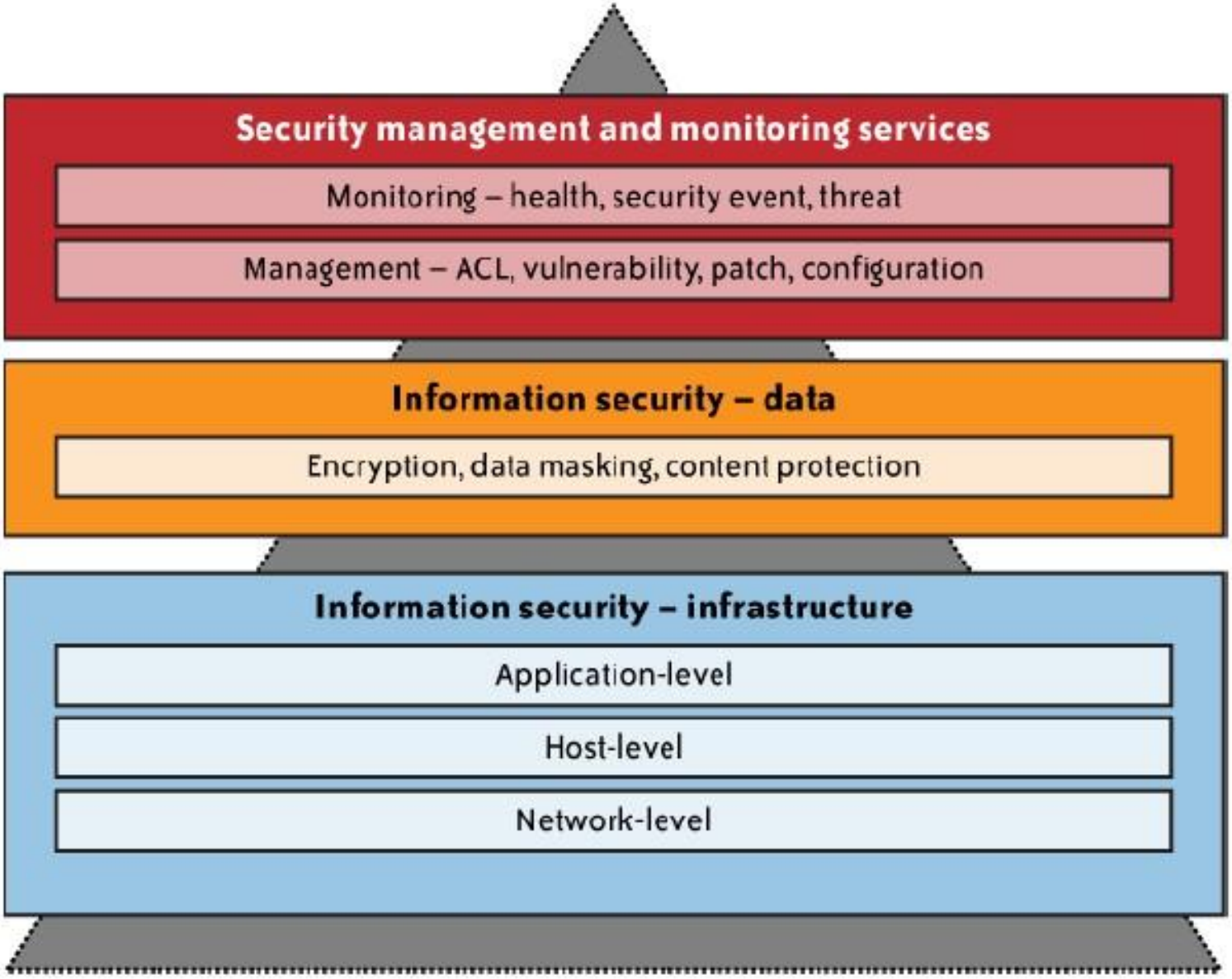
IAM Architecture & Practices

- IAM processes can be broadly classified as:
  - User Management
  - Authentication Management
  - Authorization Management
  - Access Management
  - Data Management & Provisioning
  - Monitoring & Auditing
- Current Standards & Specifications
  - Security Assertion Markup Language (SAML)
  - Service Provisioning Markup Language (SPML)
  - eXensible Access Control Markup Language (XACML)
  - Open Authentication (OAuth)

Example of SSO



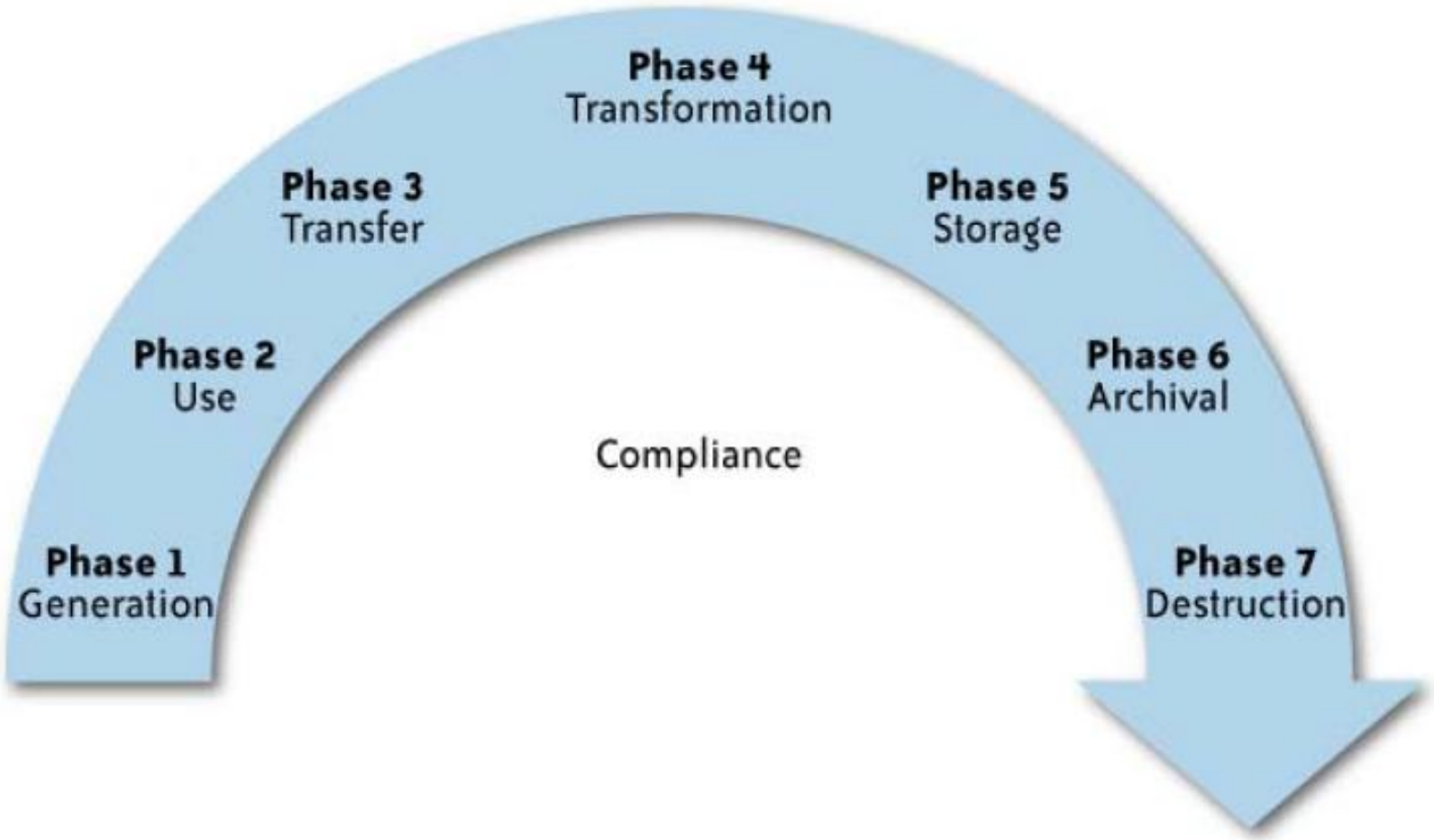
Security Management in the Cloud



Privacy

- The concept of privacy varies widely among (and also within) countries, cultures, and jurisdictions.
- Privacy is related to the collection, use, disclosure, storage, and destruction of Personally Identifiable Information (PII)
- *"The rights and obligations of individuals and organizations with respect to the collection, use, retention, and disclosure of personal information"*
- Privacy is NOT a subset of Data Security.
- You can have security and not have privacy, but you cannot have privacy without security.

Data Life Cycle



The 7 Privacy Concerns in the Cloud

1. Access
   - Data subjects have a right to know what PII is collected and can make a request to stop processing it.
2. Compliance
   - What are the privacy compliance regulations in the cloud?
3. Storage
   - Where is the PII data stored in the cloud?
   - Was it transferred to another data center in another country?
   - Is it commingled with information from other organizations that use the same Cloud Service Provider?
4. Retention
   - How long is PII transferred to the cloud retained?
   - Which retention policy governs the data?
   - Does the organization own the data, or the CSP?

The 7 Privacy Concerns in the Cloud

5. Destruction
   - How does the cloud provider destroy PII at the end of the retention period?
   - How do organizations ensure that their PII is destroyed by the CSP at the right point and is not available to other cloud users?
   - How do they know that the CSP didn't retain additional copies?
6. Audit & Monitoring
   - How can organizations provide assurance to its stakeholders that privacy requirements are duly met when their PII is stored in the cloud?
7. Privacy Breaches
   - How to know that a breach has occurred?
   - How to ensure that the CSP notifies when a breach occurs?
   - Who is responsible for managing the breach notification process?
   - If contracts include liability for breaches resulting from negligence of the CSP, how is the contract enforced and how is it determined who is at fault?

Thank You
&
Good Luck