

UNIT -I

GRID COMPUTING

Introduction to Grid COMPUTING

Why Discuss Architecture?

- Descriptive
 - Provide a common vocabulary for use when describing Grid systems
- Guidance
 - Identify key areas in which services are required
- Prescriptive
 - Define standard protocols and APIs to facilitate creation of interoperable Grid systems and portable applications

Grid Computing

- **Grid computing** is a form of distributed computing whereby a "super and virtual computer" is composed of a cluster of networked, loosely coupled computers, acting in concert to perform very large tasks.
- Grid computing (Foster and Kesselman, 1999) is a growing technology that facilitates the executions of large-scale resource intensive applications on geographically distributed computing resources.
- Facilitates flexible, secure, coordinated large scale resource sharing among dynamic collections of individuals, institutions, and resource
- Enable communities ("virtual organizations") to share geographically distributed resources as they pursue common goals
- Ian Foster and Carl Kesselman

Criteria for a Grid:

- ❖ Coordinates resources that are not subject to centralized control.
- ❖ Uses standard, open, general-purpose protocols and interfaces.
- ❖ Delivers nontrivial qualities of service.

Benefits

- ❖ Exploit Underutilized resources
- ❖ Resource load Balancing
- ❖ Virtualize resources across an enterprise
- ❖ Data Grids, Compute Grids
- ❖ Enable collaboration for virtual organizations

Elements of Grid Computing

- 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

Virtual Organizations

- A set of individuals and/or institutions defined by a set of sharing rules
- The sharing is highly controlled, with resource providers and consumers defining clearly and carefully just what is shared

An example: the set of application service providers, storage service providers, cycle providers and consultants engaged by a car manufacturer to plan for a new factory

Another example: industrial consortium building a new aircraft

Grid Applications

Data and computationally intensive applications:

This technology has been applied to computationally-intensive scientific, mathematical, and academic problems like drug discovery, economic forecasting, seismic analysis back office data processing in support of e-commerce

- A chemist may utilize hundreds of processors to screen thousands of compounds per hour.
- Teams of engineers worldwide pool resources to analyze terabytes of structural data.
- Meteorologists seek to visualize and analyze petabytes of climate data with enormous computational demands.

Resource sharing

- Computers, storage, sensors, networks, ...
- Sharing always conditional: issues of trust, policy, negotiation, payment, ...

Coordinated problem solving

- distributed data analysis, computation, collaboration, ...

Computational Grid

“A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities.”

“The Grid: Blueprint for a New Computing Infrastructure”,
Kesselman & Foster

Example : Science Grid (US Department of Energy)

Data Grid

- A **data grid** is a grid computing system that deals with data — the **controlled sharing and management of large amounts of distributed data**.
- Data Grid is the storage component of a grid environment. Scientific and engineering applications require access to large amounts of data, and often this data is widely distributed. A data grid provides seamless access to the local or remote data required to complete compute intensive calculations.

Example :

Biomedical informatics Research Network (BIRN),
the Southern California earthquake Center (SCEC).

Introduction to Grid Architecture

The nature of grid architecture

- A grid architecture identifies fundamental system components, specifies the purpose and function of these components, and indicate how these components interact.

Grid architecture requirements

The components are

- numerous
- owned and managed by different, potentially mutually distrustful organisations and individuals
may be potentially faulty
- have different security requirements and policies
heterogeneous
- connected by heterogeneous, multilevel networks
have different resource management policies
are likely to be geographically separated

Layered Grid Architecture: Fabric Layer

- Just what you would expect: the diverse mix of resources that may be shared
 - Individual computers, Condor pools, file systems, archives, metadata catalogs, networks, sensors, etc., etc.
- Defined by interfaces, not physical characteristics

Layered Grid Architecture:Connectivity Layer Communication

Internet protocols: IP, DNS, routing, etc.

Security: Grid Security Infrastructure (GSI)

Uniform authentication, authorization, and message protection mechanisms in multi-institutional setting

Single sign-on, delegation, identity mapping

Public key technology, SSL, X.509, GSS-API

Supporting infrastructure: Certificate Authorities, certificate & key management, ...

Layered Grid Architecture:Resource Layer

Grid Resource Allocation Mgmt (GRAM)

Remote allocation, reservation, monitoring, control of compute resources

GridFTP protocol (FTP extensions)

High-performance data access & transport

Grid Resource Information Service (GRIS)

Access to structure & state information

Network reservation, monitoring, control

All built on connectivity layer: GSI & IP

Layered Grid Architecture: Collective Layer
Index servers aka metadirectory services
Custom views on dynamic resource collections
assembled by a community
Resource brokers (e.g., Condor Matchmaker)
Resource discovery and allocation
Replica catalogs
Replication services
Co-reservation and co-allocation services
Workflow management services
Etc.

Layered Grid Architecture: Applications layer

There are user applications that operate within the VO environment

Applications are constructed by calling upon services defined at any layer

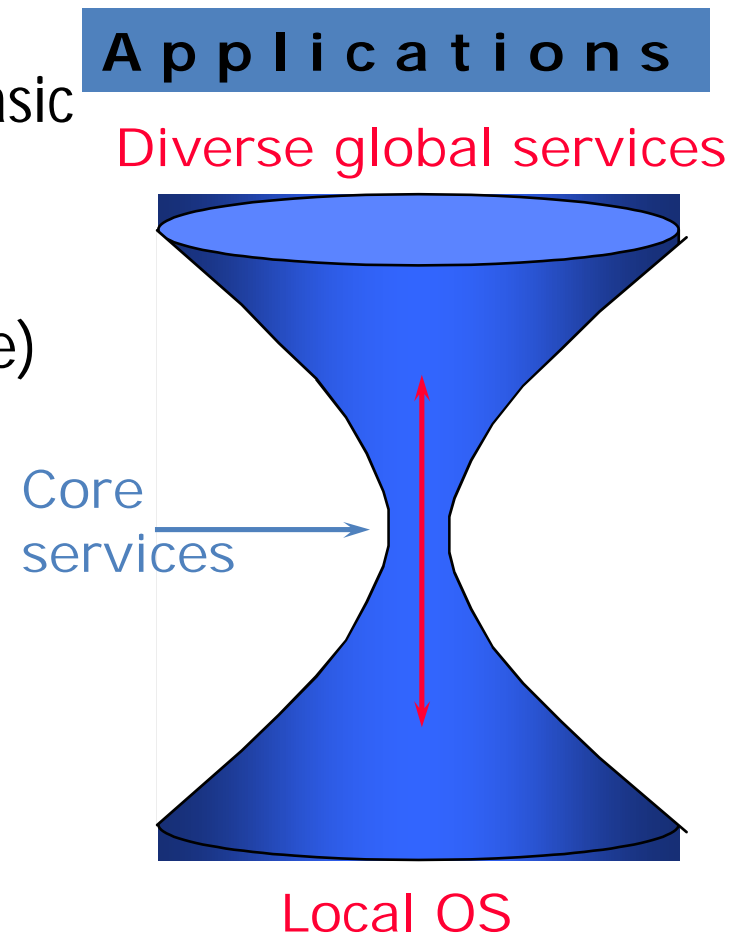
Each of the layers are well defined using protocols, provide access to services

Well-defined APIs also exist to work with these services

Grid Architecture

The Hourglass Model

- Focus on architecture issues
 - Propose set of core services as basic infrastructure
 - Used to construct high-level, domain-specific solutions (diverse)
- Design principles
 - Keep participation cost low
 - Enable local control
 - Support for adaptation
 - “IP hourglass” model



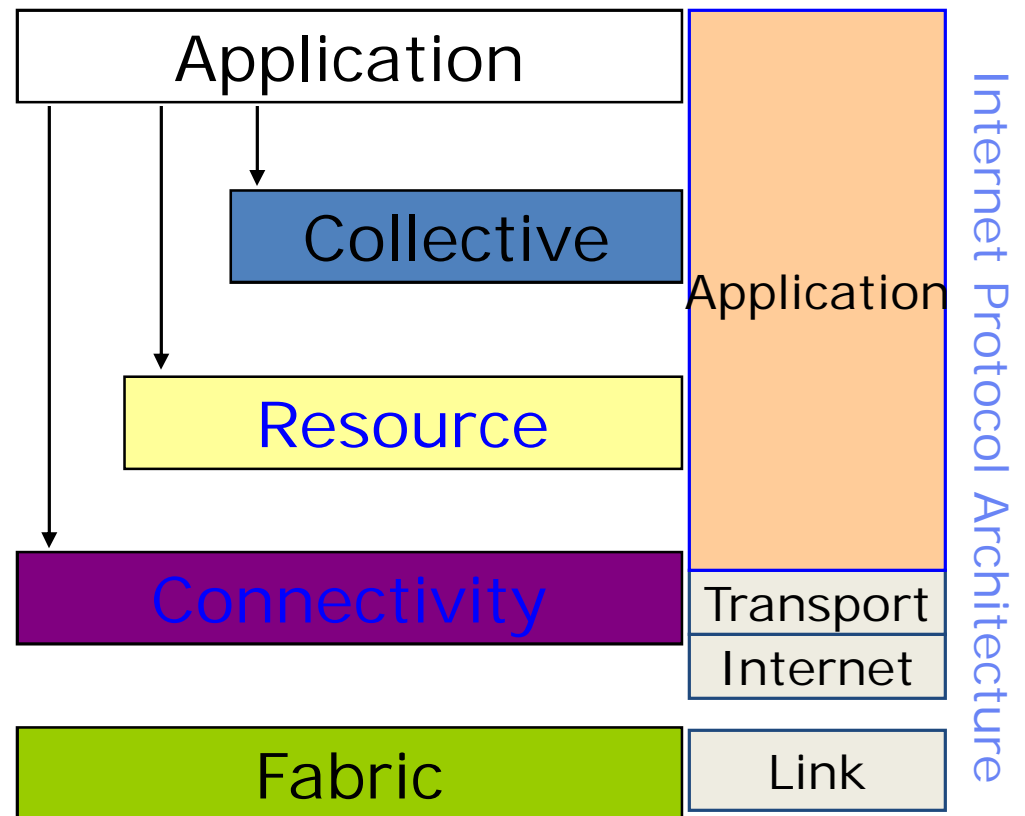
Layered Grid Architecture (By Analogy to Internet Architecture)

“Coordinating multiple resources”:
ubiquitous infrastructure services,
app-specific distributed services

“Sharing single resources”:
negotiating access, controlling use

“Talking to things”:
communication (Internet protocols) & security

“Controlling things locally”:
Access to, & control of, resources



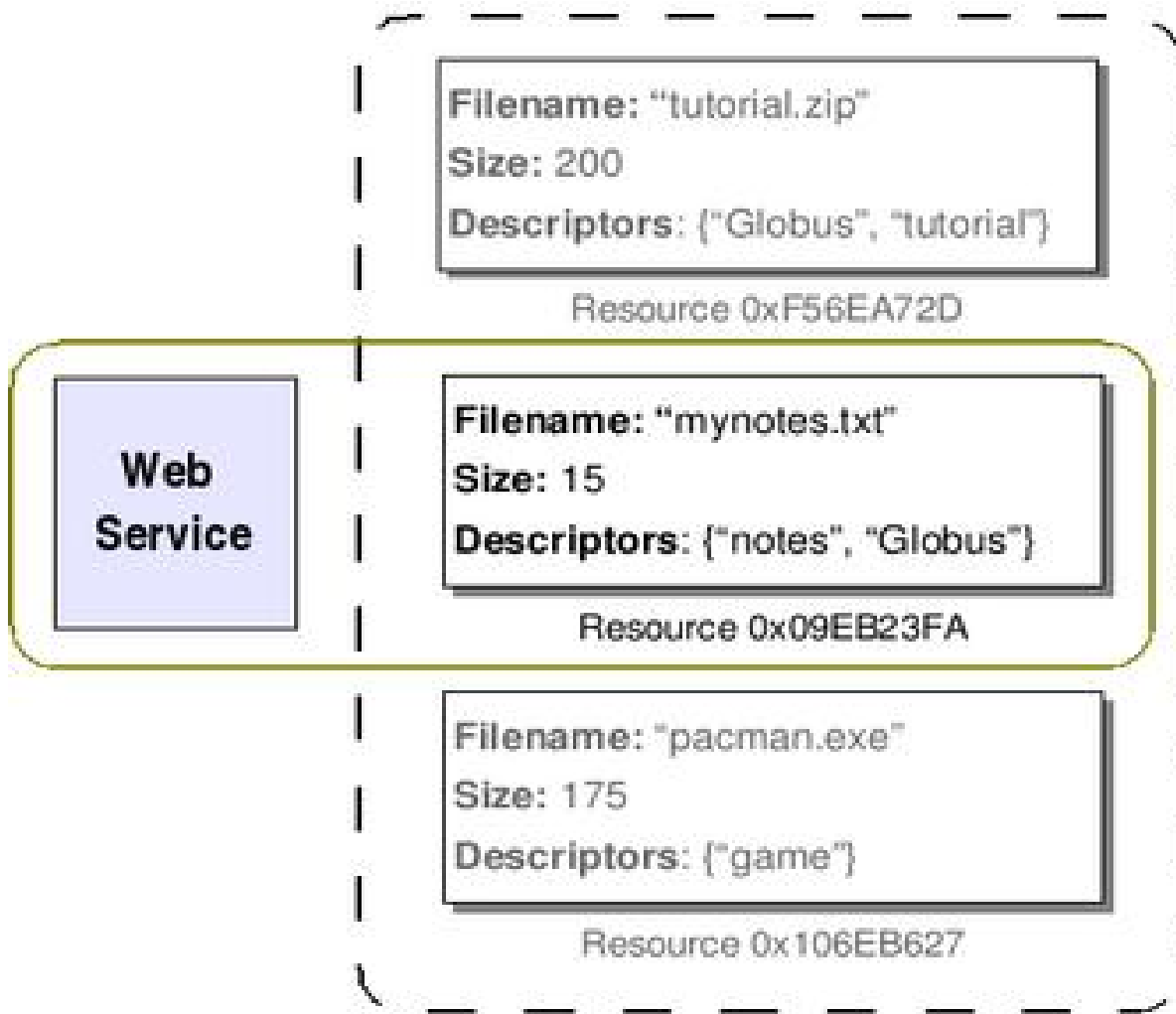
Example: Data Grid Architecture

App	Discipline-Specific Data Grid Application
Collective (App)	Coherency control, replica selection, task management, virtual data catalog, virtual data code catalog, ...
Collective (Generic)	Replica catalog, replica management, co-allocation, certificate authorities, metadata catalogs,
Resource	Access to data, access to computers, access to network performance data, ...
Connect	Communication, service discovery (DNS), authentication, authorization, delegation
Fabric	Storage systems, clusters, networks, network caches, ...

Web services and the Grid

GT4 replaced OGSI by WSRF (Web Service Resource Framework)
Framework developed as a joint effort of W3C and OGF groups
GWSDL foi abandonada

RESOURCES



$$\begin{aligned} &\text{Web Service} \\ &+ \\ &\text{Resource} \\ &= \\ &\text{WS-Resource} \end{aligned}$$

UNIT II-CLOUD ARCHITECTURE AND MODEL

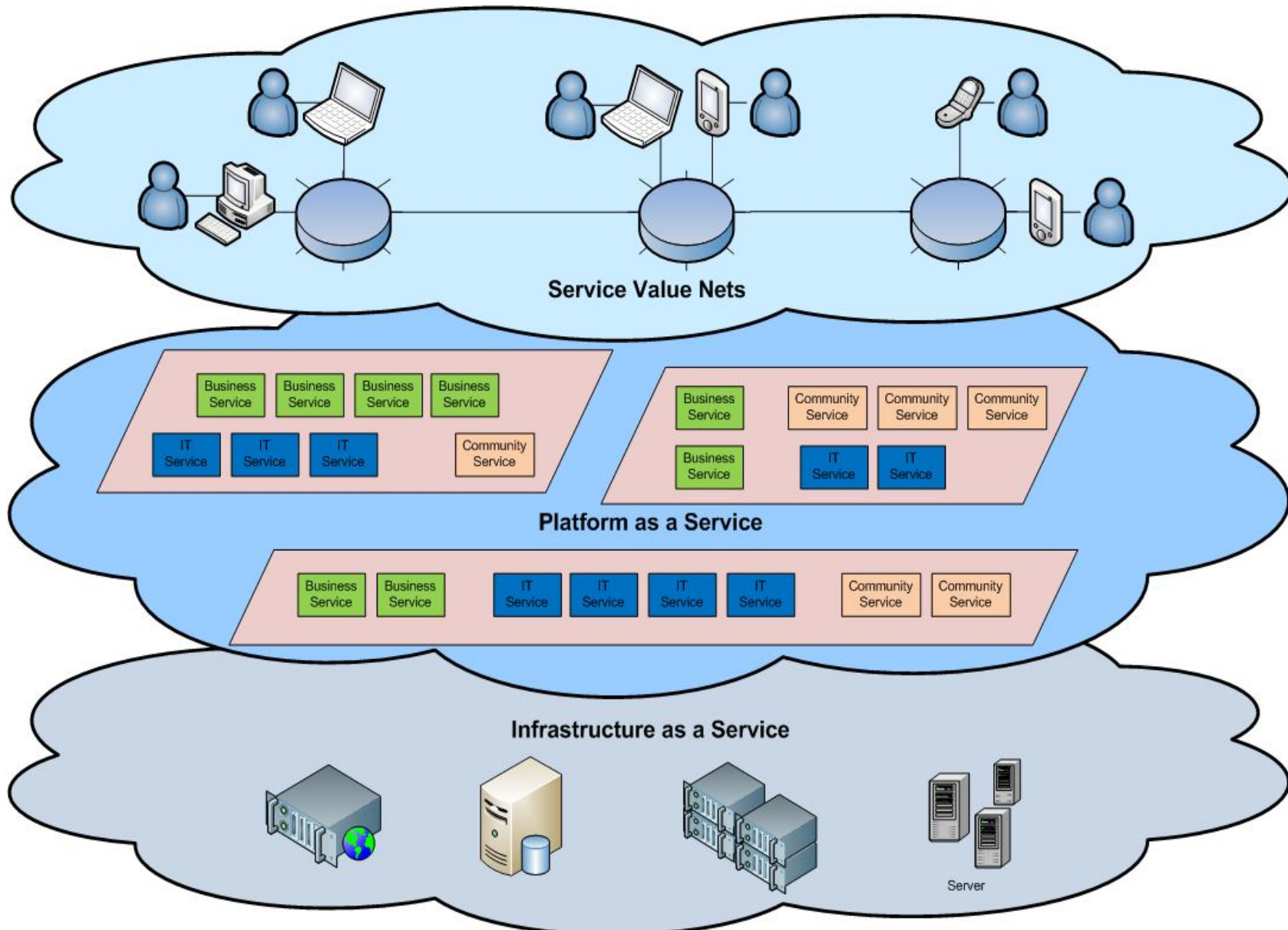
What is Cloud Computing?

- **Cloud Computing** is a general term used to describe a new class of network based computing that takes place over the Internet,
 - basically a step on from Utility Computing
 - a collection/group of integrated and networked hardware, software and Internet infrastructure (called a platform).
 - Using the Internet for communication and transport provides hardware, software and networking services to clients
- These platforms hide the complexity and details of the underlying infrastructure from users and applications by providing very simple graphical interface or API (Applications Programming Interface).

What is Cloud Computing?

- In addition, the platform provides on demand services, that are always on, anywhere, anytime and any place.
- Pay for use and as needed, elastic
 - scale up and down in capacity and functionalities
- The hardware and software services are available to
 - general public, enterprises, corporations and businesses markets

Cloud Architecture



3 Cloud Service Models

- Cloud Software as a Service (SaaS)
 - The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure and accessible from various client devices through a thin client interface such as a Web browser (e.g., web-based email).
- Cloud Platform as a Service (PaaS)
 - The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created applications using programming languages and tools supported by the provider (e.g., Java, Python, .Net).
- Cloud Infrastructure as a Service (IaaS)
 - The capability provided to the consumer is to rent processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications.

Cloud Computing Characteristics

Common Characteristics:

Massive Scale

Resilient Computing

Homogeneity

Geographic Distribution

Virtualization

Service Orientation

Low Cost Software

Advanced Security

Essential Characteristics:

On Demand Self-Service

Broad Network Access

Rapid Elasticity

Resource Pooling

Measured Service

Cloud Service Models

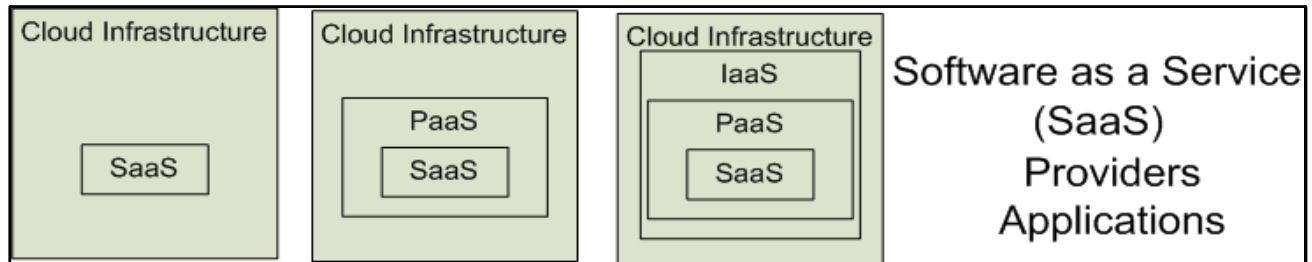
Software as a Service (SaaS)

Platform as a Service (PaaS)

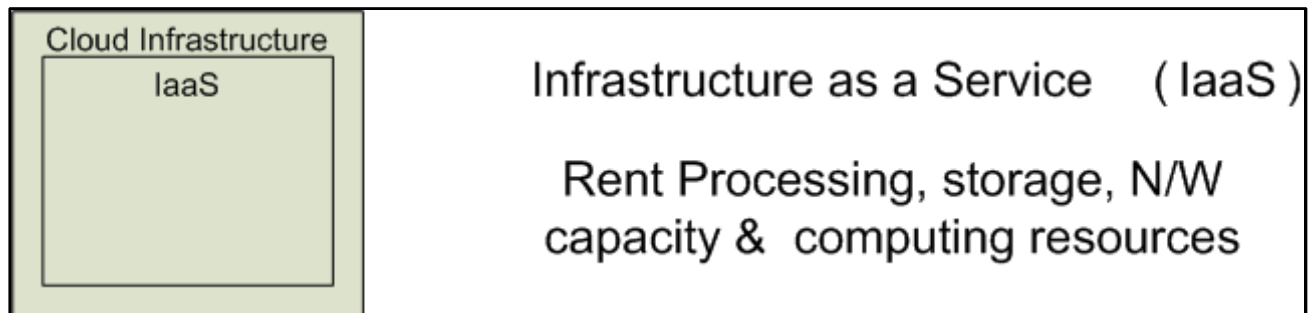
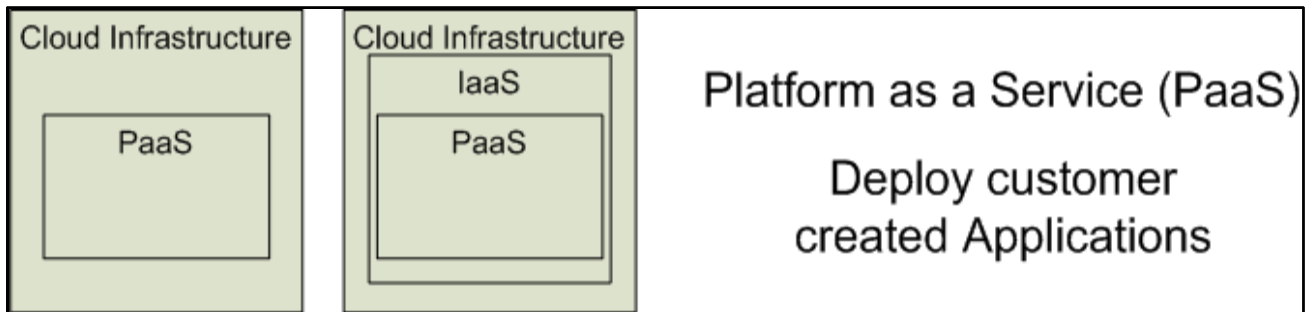
Infrastructure as a Service (IaaS)

SalesForce CRM

LotusLive



Google App Engine



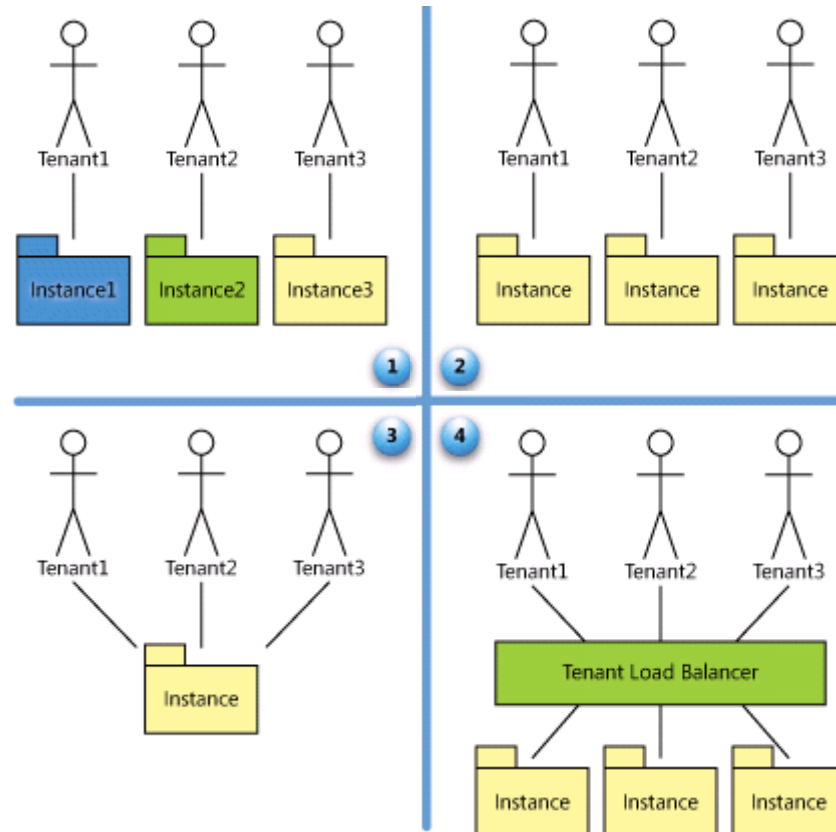
SaaS Maturity Model

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

Level 2: Configurable per
customer

Level 3: configurable &
Multi-Tenant-Efficient

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



Software as a Service (SaaS)

- SaaS is a model of software deployment where an application is hosted as a service provided to customers across the Internet.
- SaaS alleviates the burden of software maintenance/support
 - but users relinquish control over software versions and requirements.
- Terms that are used in this sphere include
 - **Platform as a Service** (PaaS) and
 - **Infrastructure as a Service** (IaaS)

Cloud Computing Service Layers

	Services	Description
Application Focused	Services	Services – Complete business services such as PayPal, OpenID, OAuth, Google Maps, Alexa
	Application	Application – Cloud based software that eliminates the need for local installation such as Google Apps, Microsoft Online
	Development	Development – Software development platforms used to build custom cloud based applications (PAAS & SAAS) such as SalesForce
Infrastructure Focused	Platform	Platform – Cloud based platforms, typically provided using virtualization, such as Amazon ECC, Sun Grid
	Storage	Storage – Data storage or cloud based NAS such as CTERA, iDisk, CloudNAS
	Hosting	Hosting – Physical data centers such as those run by IBM, HP, NaviSite, etc.

Basic Cloud Characteristics

- The “**no-need-to-know**” in terms of the underlying details of infrastructure, applications interface with the infrastructure via the APIs.
- The “**flexibility and elasticity**” allows these systems to scale up and down at will
 - utilising the resources of all kinds
 - CPU, storage, server capacity, load balancing, and databases
- The “**pay as much as used and needed**” type of utility computing and the “**always on!, anywhere and any place**” type of network-based computing.

Basic Cloud Characteristics

- Cloud are transparent to users and applications, they can be built in multiple ways
 - branded products, proprietary open source, hardware or software, or just off-the-shelf PCs.
- In general, they are built on clusters of PC servers and off-the-shelf components plus Open Source software combined with in-house applications and/or system software.

Cloud Security Challenges

- Trusting vendor's security model
- Multi-tenancy
- Data ownership issues
- QoS guarantees
- Attraction to hackers (high-value target)
- Security of virtual OSs in the cloud
- Obtaining support from cloud vendor for security related investigations

What is the purpose and benefits?

- Cloud computing enables companies and applications, which are system infrastructure dependent, to be infrastructure-less.
- By using the Cloud infrastructure on “pay as used and on demand”, all of us can save in capital and operational investment!
- Clients can:
 - Put their data on the platform instead of on their own desktop PCs and/or on their own servers.
 - They can put their applications on the cloud and use the servers within the cloud to do processing and data manipulations etc.

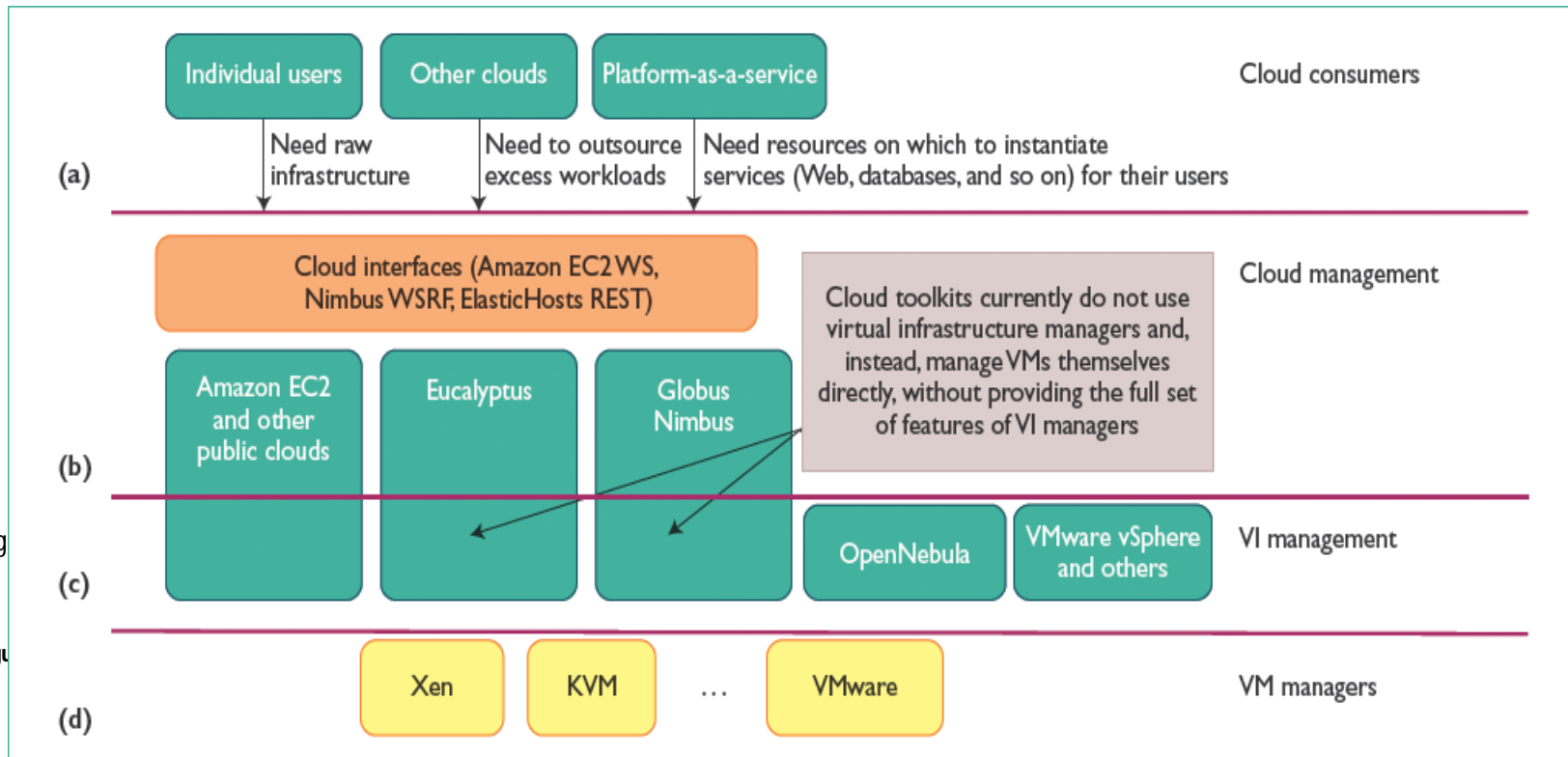
Cloud-Sourcing

- Why is it becoming a Big Deal:
 - Using high-scale/low-cost providers,
 - Any time/place access via web browser,
 - Rapid scalability; incremental cost and load sharing,
 - Can forget need to focus on local IT.
- Concerns:
 - Performance, reliability, and SLAs,
 - Control of data, and service parameters,
 - Application features and choices,
 - Interaction between Cloud providers,
 - No standard API – mix of SOAP and REST!
 - Privacy, security, compliance, trust...

Opportunities and Challenges

- The use of the cloud provides a number of opportunities:
 - It enables services to be used without any understanding of their infrastructure.
 - Cloud computing works using economies of scale:
 - It potentially lowers the outlay expense for start up companies, as they would no longer need to buy their own software or servers.
 - Cost would be by on-demand pricing.
 - Vendors and Service providers claim costs by establishing an ongoing revenue stream.
 - Data and services are stored remotely but accessible from “anywhere”.

Cloud Ecosystem



Fig

Fig

Cloud Ecosystem

- The public cloud ecosystem has evolved around providers, users, and technologies.
- The previous figure suggests one possible ecosystem for private clouds. There are 4 levels of development of ecosystem development: cloud users/consumers, cloud management, VI management, and VM managers.
- At the cloud management level, the cloud manager provides virtualized resources over an IaaS platform.
- At the virtual infrastructure (VI) management level, the manager allocates VMs over multiple server clusters. Examples: OpenNebula, VMWare vSphere. These can manage VM managers like Xen, KVM etc. These support dynamic placement and VM management on a pool of physical resources, automatic load balancing, server consolidation, and dynamic infrastructure resizing and partitioning.
- Finally, at the VM management level the VM managers handles VMs installed on individual host machines. Examples: Xen, VMWare, KVM.

Advantages of Cloud Computing

- Lower computer costs:
 - You do not need a high-powered and high-priced computer to run cloud computing's web-based applications.
 - Since applications run in the cloud, not on the desktop PC, your desktop PC does not need the processing power or hard disk space demanded by traditional desktop software.
 - When you are using web-based applications, your PC can be less expensive, with a smaller hard disk, less memory, more efficient processor...
 - In fact, your PC in this scenario does not even need a CD or DVD drive, as no software programs have to be loaded and no document files need to be saved.

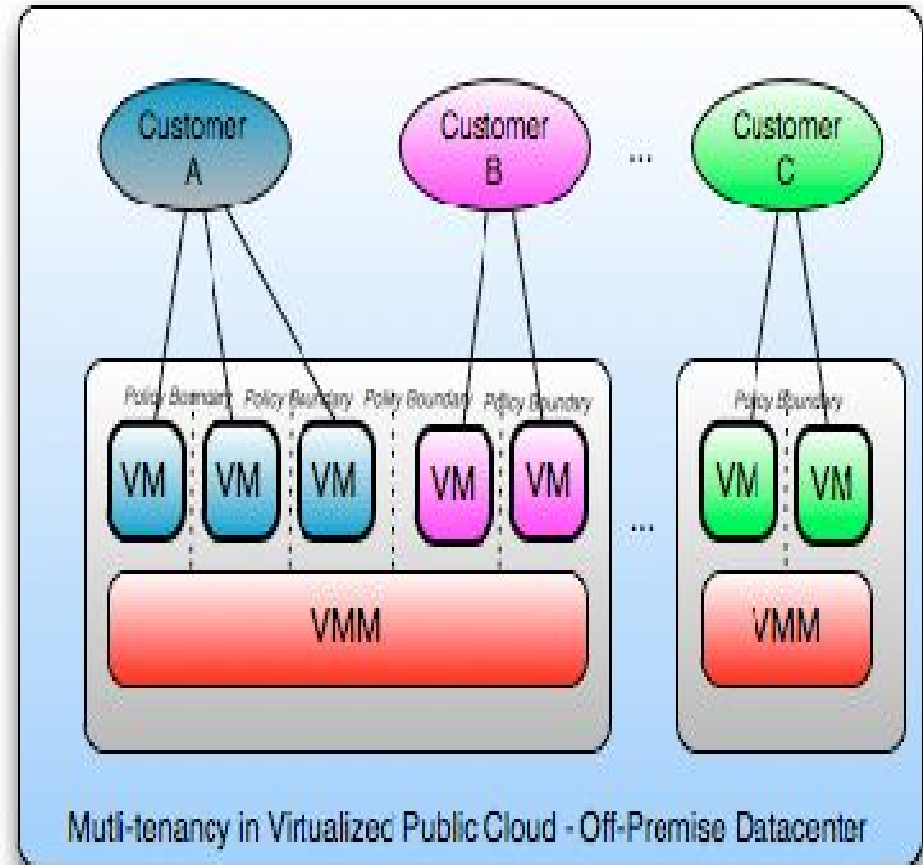
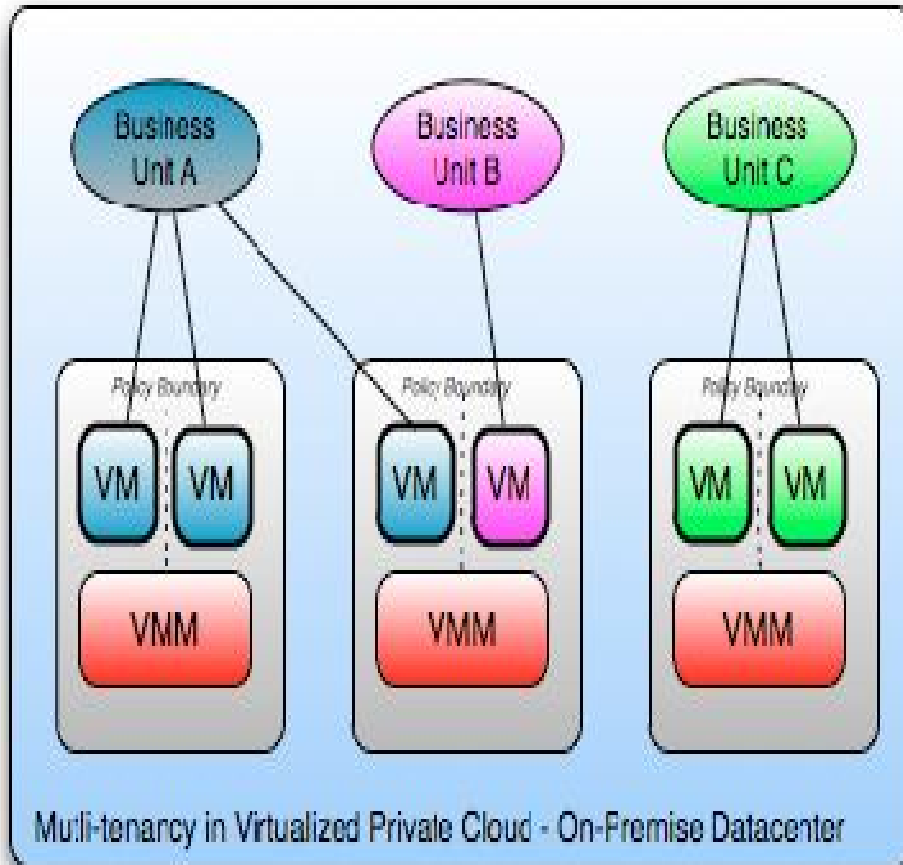
Advantages of Cloud Computing

- Improved performance:
 - With few large programs hogging your computer's memory, you will see better performance from your PC.
 - Computers in a cloud computing system boot and run faster because they have fewer programs and processes loaded into memory...
- Reduced software costs:
 - Instead of purchasing expensive software applications, you can get most of what you need for free-ish!
 - most cloud computing applications today, such as the Google Docs suite.
 - better than paying for similar commercial software
 - which alone may be justification for switching to cloud applications.

Disadvantages of Cloud Computing

- Can be slow:
 - Even with a fast connection, web-based applications can sometimes be slower than accessing a similar software program on your desktop PC.
 - Everything about the program, from the interface to the current document, has to be sent back and forth from your computer to the computers in the cloud.
 - If the cloud servers happen to be backed up at that moment, or if the Internet is having a slow day, you would not get the instantaneous access you might expect from desktop applications.

Private VS Public Cloud



Private Cloud of Company XYZ with 3 business units, each with different security, SLA, governance and chargeback policies on shared infrastructure

Public Cloud Provider with 3 business customers, each with different security, SLA, governance and billing policies on shared infrastructure

UNIT-III -CLOUD INFRASTRUCTURE

Introduction

High performance networks and advanced development of internet is the basis for cloud computing .

Cloud computing has started taking shape incorporating virtualization and on demand deployment and internet delivery of services

Conventional Computing vs. Cloud Computing

Conventional

- Manually Provisioned
- Dedicated Hardware
- Fixed Capacity
- Pay for Capacity
- Capital & Operational Expenses

Cloud

- Self-provisioned
- Shared Hardware
- Elastic Capacity
- Pay for Use
- Operational Expenses

Five Key Cloud Attributes:

1. Shared / pooled resources
2. Broad network access
3. On-demand self-service
4. Scalable and elastic
5. Metered by use

Shared / Pooled Resources:

- Resources are drawn from a common pool
- Common resources build economies of scale
- Common infrastructure runs at high efficiency

Broad Network Access:

- Open standards and APIs
- Almost always IP, HTTP, and REST
- Available from anywhere with an internet connection

On-Demand Self-Service:

- Completely automated
- Users abstracted from the implementation
- Near real-time delivery (seconds or minutes)
- Services accessed through a self-serve web interface

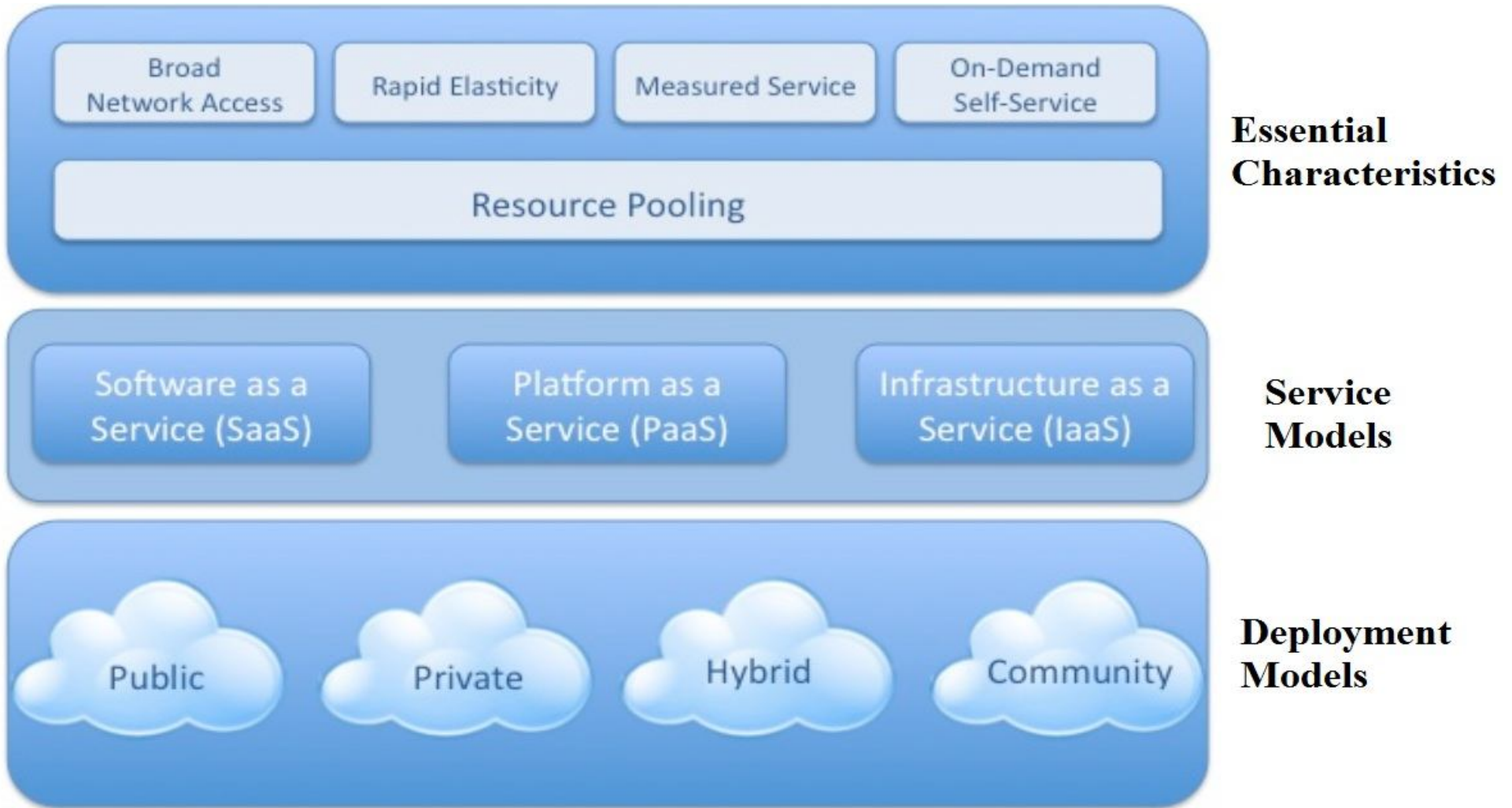
Scalable and Elastic:

- Resources dynamically-allocated between users
- Additional resources dynamically-released when needed
- Fully automated

Metered by Use:

- Services are metered, like a utility
- Users pay only for services used
- Services can be cancelled at any time

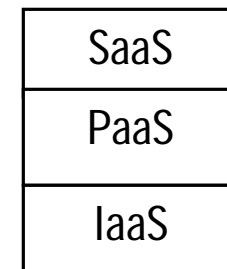
Architecture Overview



Architectural Layers of Cloud Computing

In the cloud computing stack, there are three basic layers that together create cloud environment. They are:

1. Infrastructure as a Service (IaaS)
2. Platform as a Service (PaaS)
3. Software as a Service (SaaS)

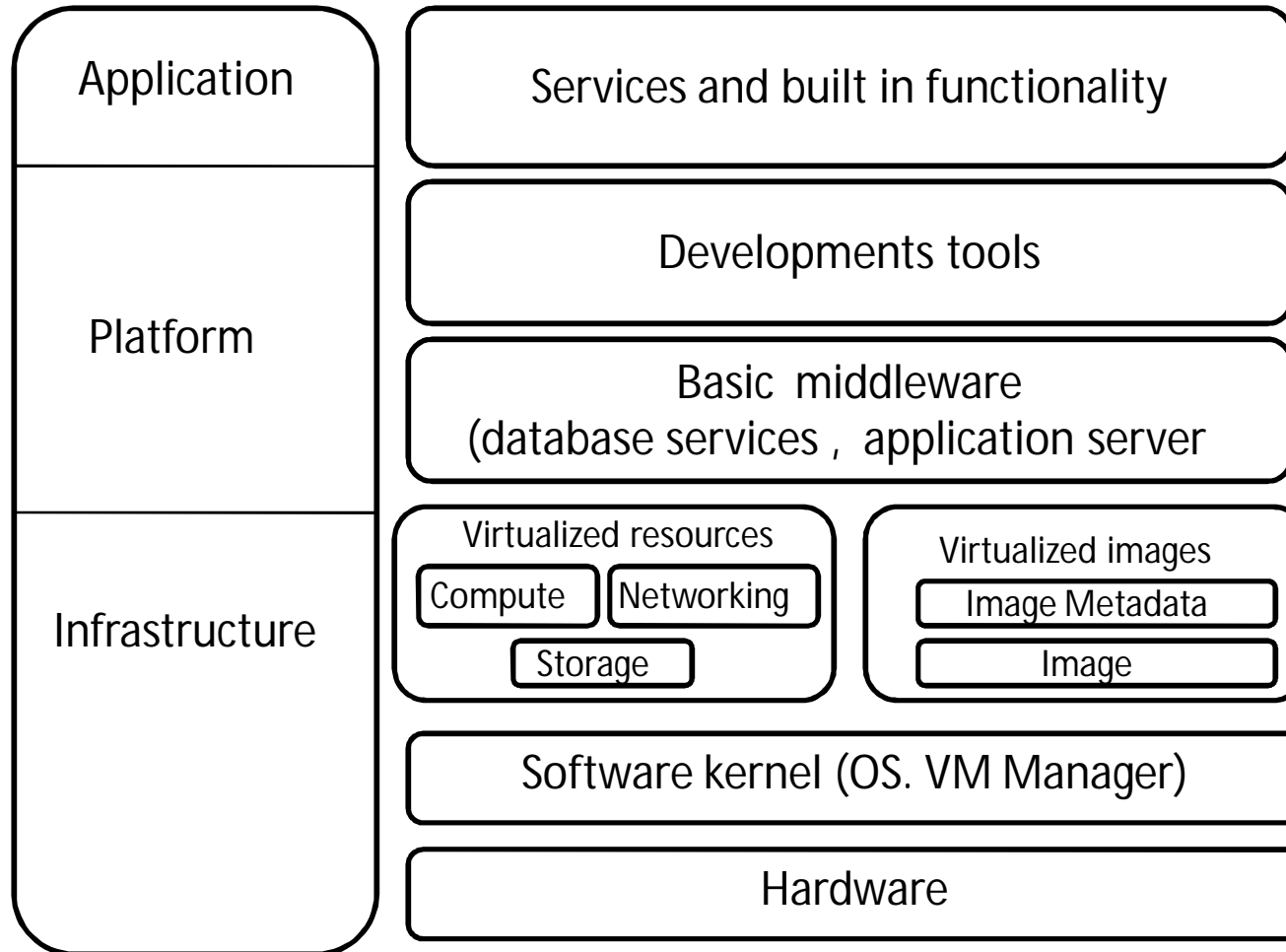


Service Delivery Model Examples

	Amazon	Google	Microsoft	Salesforce
SaaS				
PaaS				
IaaS				

Products and companies shown for illustrative purposes only and should not be construed as an endorsement

Framework of cloud computing



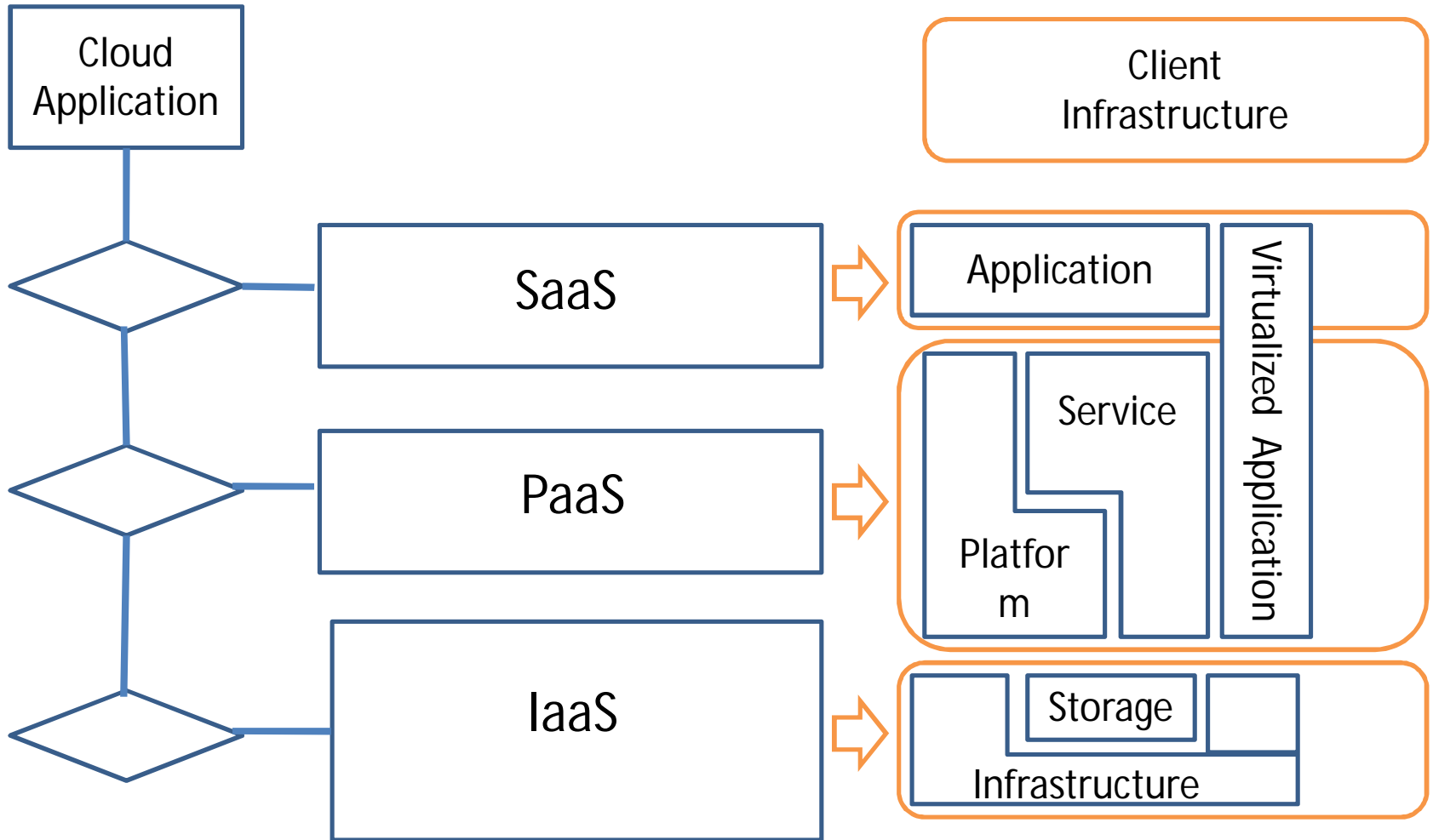
Virtual infrastructure management and Cloud Computing

- For building the cloud environment a variety of requirements must be met to provide a uniform and homogeneous view of the virtualized resources.
- Virtual Infrastructure Management is the key component to build the cloud environment which does the dynamic orchestration of virtual machines on a pool of physical resources.

Virtual infrastructure management and Cloud Computing

- Virtual infrastructure management provide primitives to schedule and manage VMs across multiple physical hosts.
- Cloud management provide remote and secure interface for creating controlling and monitoring virtualized resources on IaaS

View of Cloud Deployment



Software as a Service

❖ It is a Deployment/Delivery model

- Hosted and managed by vendor
- Delivered across the internet

❖ It is a Business Model : usage-based pricing(vs. perpetual license model of on -premise software).Examples:

- Per user per month
- Per transaction
- Per GB of storage per month

Software as a Service

Architectural

- Multi-tenancy
- Scalability
- Security
- Performance

Functional

- Provisioning
- Billing
- Metering
- Monitoring

Service Deployment Methodology

- It is paramount that IT and business goals are aligned throughout the process when considering a move to cloud computing, such as cost savings, security, control, flexibility, manageability, simplification, ease of use, expandability, reliability, availability...



Assessment and Design

Proper alignment with business and technical goals

- **Cloud Assessment and Design**
Working with business users and IT professionals to define high-level requirements (Business Driver)
Assessing the Pros and Cons for using Cloud solutions
Determining appropriate risks and management strategies for Cloud solutions
- **Cloud Solution Selection**
Determining specific business and technical challenges
Choosing the right Cloud alternatives (type and delivery model)
Identifying the management requirements for the different Cloud alternatives
Defining the solution alternatives and the merits / risks with each

Deployment and Migration

Assessment and Design leads to a working solutions document (published best practice solutions guides)

- Solutions planning
- Investment planning & acquisition
- Integration & test
- Deployment, documentation, operations & maintenance

Monitoring and Tuning

Effectively Monitoring Your Cloud Ecosystem

- A cloud monitoring solution should identify problems before they become critical and adapt as business requirements change. A nice option may be to deploy a third party monitoring service to ensure customer satisfaction and allow an unbiased perspective on application performance. By implementing a comprehensive monitoring solution IT organization are equipped with the tools to determine real business value for cloud solutions and to provide an important feedback mechanism for tuning their cloud solutions.

UNIT-IV PROGRAMMING MODEL

New Cloud Programming Paradigms

Easy to write and run highly parallel programs in new cloud programming paradigms:

- Google: MapReduce and Sawzall
- Amazon: Elastic MapReduce service (pay-as-you-go)
- Google (MapReduce)
 - Indexing: a chain of 24 MapReduce jobs
 - ~200K jobs processing 50PB/month (in 2006)
- Yahoo! (Hadoop + Pig)
 - WebMap: a chain of 100 MapReduce jobs
 - 280 TB of data, 2500 nodes, 73 hours
- Facebook (Hadoop + Hive)
 - ~300TB total, adding 2TB/day (in 2008)
 - 3K jobs processing 55TB/day
- Similar numbers from other companies, e.g., Yieldex, eharmony.com, etc.
- NoSQL: MySQL is an industry standard, but Cassandra is 2400 times faster!

What is MapReduce?

- Terms are borrowed from Functional Language (e.g., Lisp)

Sum of squares:

- `(map square '(1 2 3 4))`

- Output: (1 4 9 16)

- [processes each record sequentially and independently]

- `(reduce + '(1 4 9 16))`

- `(+ 16 (+ 9 (+ 4 1)))`

- Output: 30

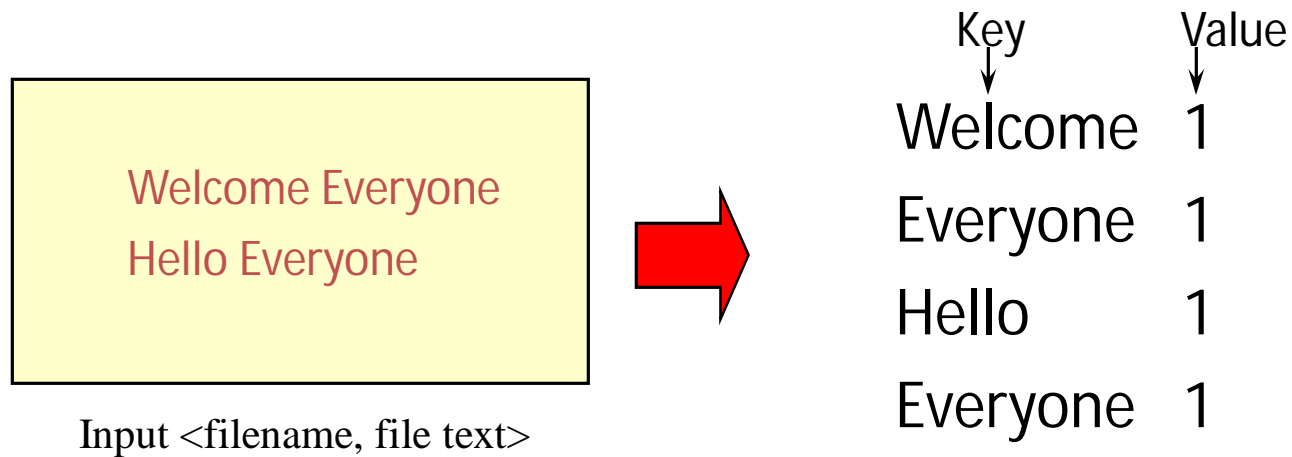
- [processes set of all records in batches]

- Let's consider a sample application: Wordcount

- You are given a huge dataset (e.g., Wikipedia dump) and asked to list the count for each of the words in each of the documents therein

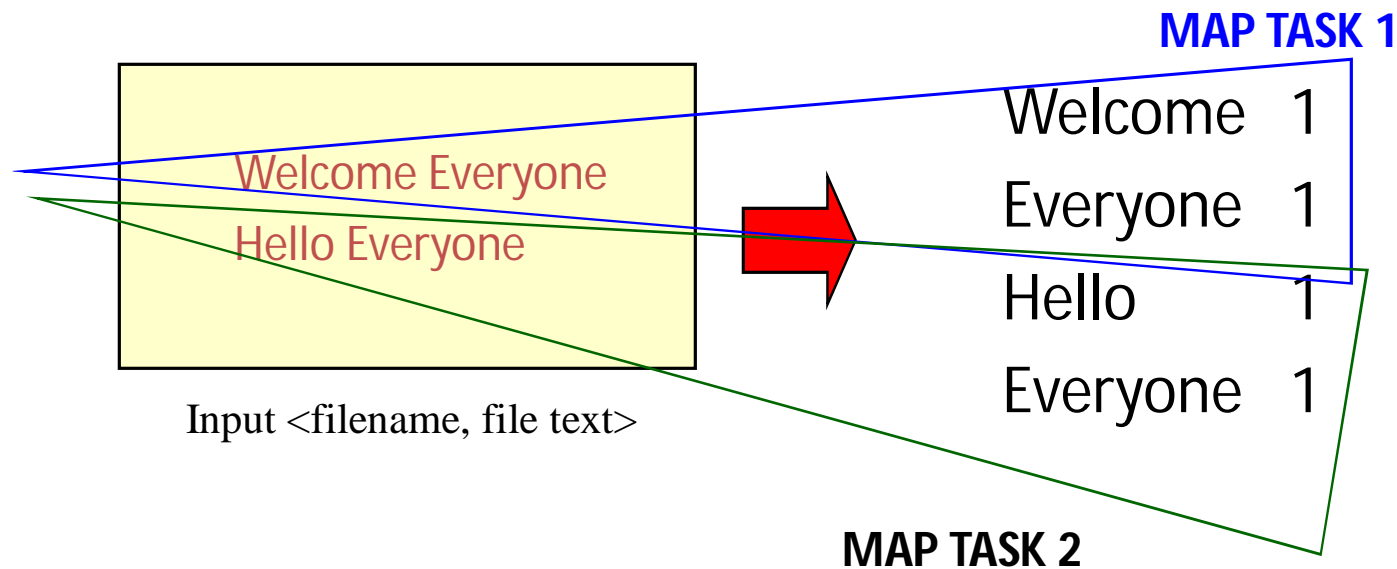
Map

- Process individual records to generate intermediate key/value pairs.



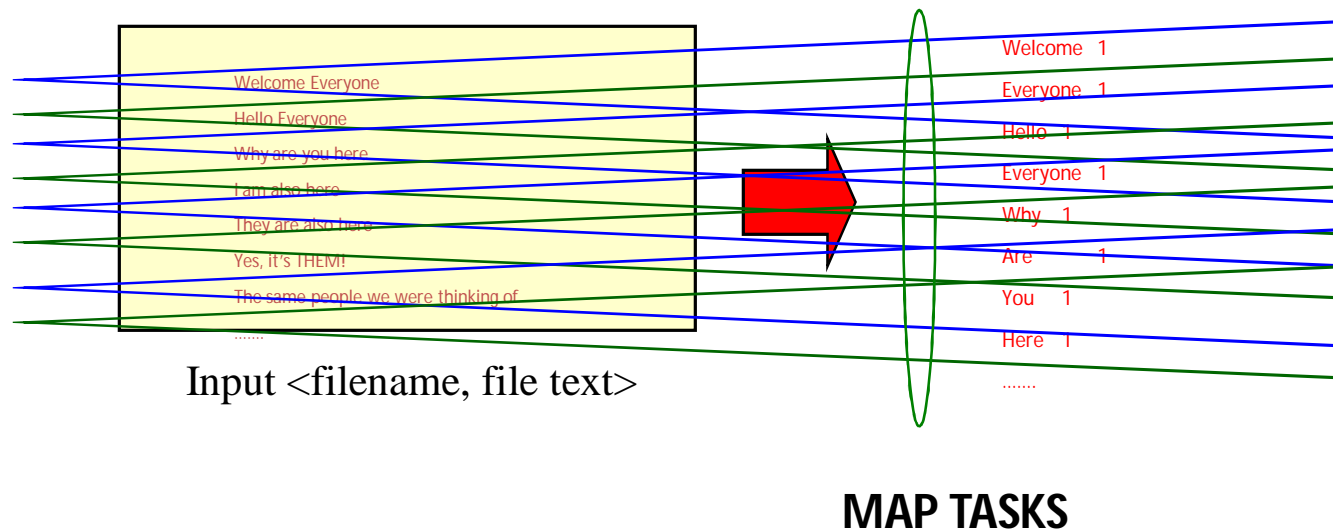
Map

- Parallelly Process individual records to generate intermediate key/value pairs.



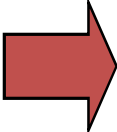
Map

- Parallely Process a large number of individual records to generate intermediate key/value pairs.



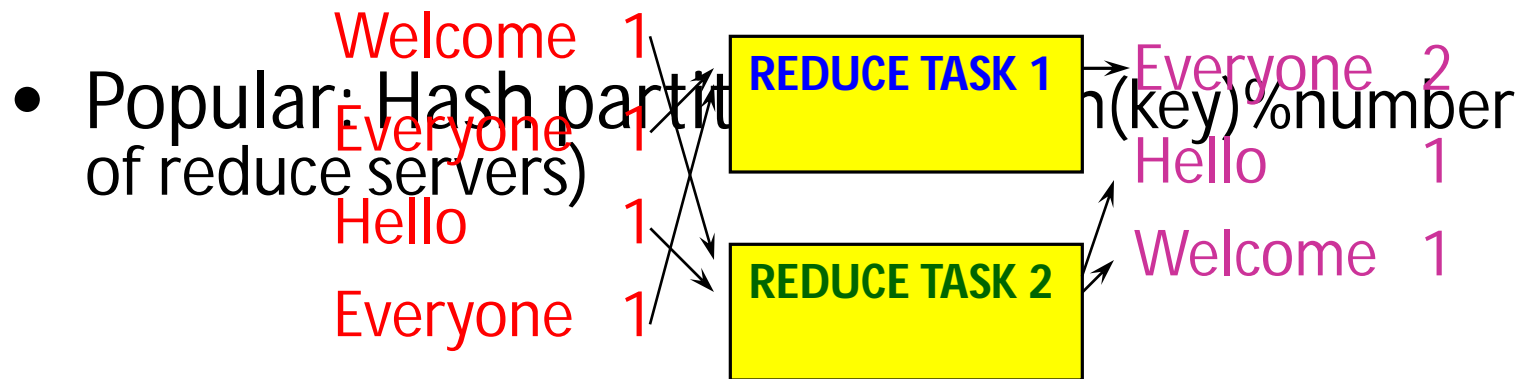
Reduce

- Reduce processes and merges all intermediate values associated per key

			Key	Value
			↓	↓
Welcome	1		Everyone	2
Everyone	1		Hello	1
Hello	1		Welcome	1
Everyone	1			

Reduce

- Each key assigned to one Reduce
- Parallelly Processes and merges all intermediate values by partitioning keys



- Input: (lineNumber, line) records
- Output: lines matching a given pattern
- Map:

```
if(line matches pattern):  
    output(line)
```
- Reduce: identify function
 - Alternative: no reducer (map-only job)

- **Input:** (key, value) records
- **Output:** same records, sorted by key

- **Map:** identity function
- **Reduce:** identity function

- **Trick:** Pick partitioning function h such that $k_1 < k_2 \Rightarrow h(k_1) < h(k_2)$

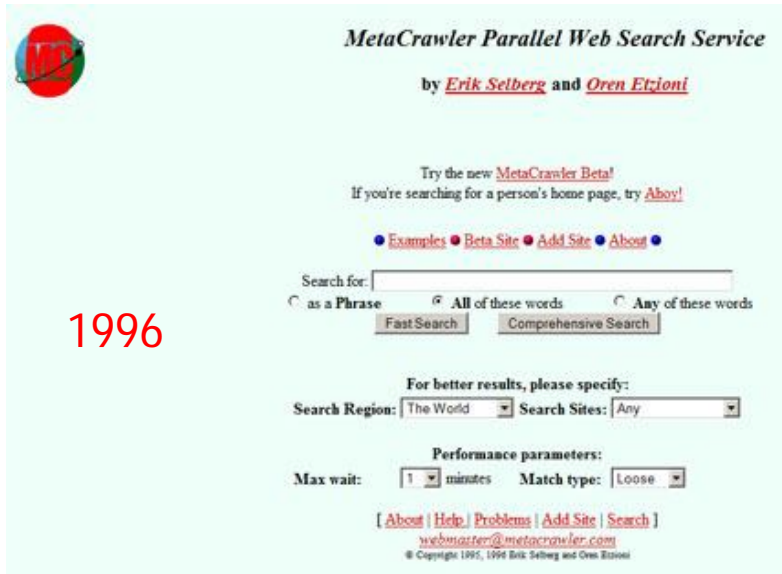
- **Input:** (filename, text) records
- **Output:** list of files containing each word
- **Map:**
 - foreach** word **in**
 - text.split():
 - output(word, filename)
- **Combine:** uniquify filenames for each word

- **Input:** (filename, text) records
- **Output:** top 100 words occurring in the most files
- Two-stage solution:
 - **Job 1:**
 - Create inverted index, giving (word, list(file)) records
 - **Job 2:**
 - Map each (word, list(file)) to (count, word)
 - Sort these records by count as in sort job

What is Hadoop?

- Apache top level project, open-source implementation of frameworks for reliable, scalable, distributed computing and data storage.
- It is a flexible and highly-available architecture for large scale computation and data processing on a network of commodity hardware.

Search engines in 1990s



1996



1996



1996



1997

[New Search](#) • [TopNews](#) • [Sites by Subject](#) • [Top 5% Sites](#) • [City Guide](#) • [Pictures & Sounds](#)
[PeopleFind](#) • [Point Review](#) • [Road Maps](#) • [Software](#) • [About Lycos](#) • [Chb Lycos](#) • [Help](#)

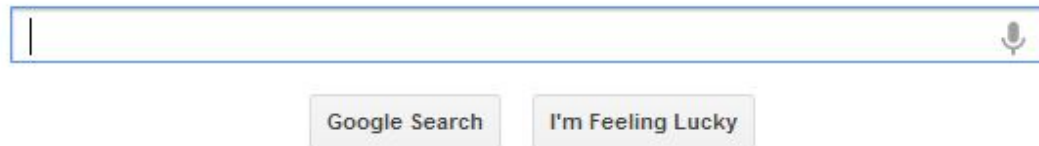
[Add Your Site to Lycos](#)

Copyright © 1996 Lycos™, Inc. All Rights Reserved.
 Lycos is a trademark of Carnegie Mellon University.
[Questions & Comments](#)

Google search engines



1998



2013

Hadoop Code - Map

```
public static class MapClass extends MapReduceBase
    implements Mapper<LongWritable, Text, Text,
        IntWritable> {
    private final static IntWritable one =
        new IntWritable(1);
    private Text word = new Text();

    public void map( LongWritable key, Text value,
        OutputCollector<Text, IntWritable> output,
        Reporter reporter)
        throws IOException {
        String line = value.toString();
        StringTokenizer itr = new StringTokenizer(line);
        while (itr.hasMoreTokens()) {
            word.set(itr.nextToken());
            output.collect(word, one);
        }
    }
}
```

Hadoop Code - Reduce

```
public static class ReduceClass extends MapReduceBase
    implements Reducer<Text, IntWritable, Text,
        IntWritable> {
    public void reduce(
        Text key,
        Iterator<IntWritable> values,
        OutputCollector<Text, IntWritable> output,
        Reporter reporter)
        throws IOException {
        int sum = 0;
        while (values.hasNext()) {
            sum += values.next().get();
        }
        output.collect(key, new IntWritable(sum));
    }
}
```

Hadoop Code - Driver

```
// Tells Hadoop how to run your Map-Reduce job
public void run (String inputPath, String outputPath)
    throws Exception {
    // The job. WordCount contains MapClass and Reduce.
    JobConf conf = new JobConf(WordCount.class);
    conf.setJobName("mywordcount");
    // The keys are words
    (strings) conf.setOutputKeyClass(Text.class);
    // The values are counts (ints)
    conf.setOutputValueClass(IntWritable.class);
    conf.setMapperClass(MapClass.class);
    conf.setReducerClass(ReduceClass.class);
    FileInputFormat.addInputPath(
        conf, newPath(inputPath));
    FileOutputFormat.setOutputPath(
        conf, new Path(outputPath));
    JobClient.runJob(conf);
}
```

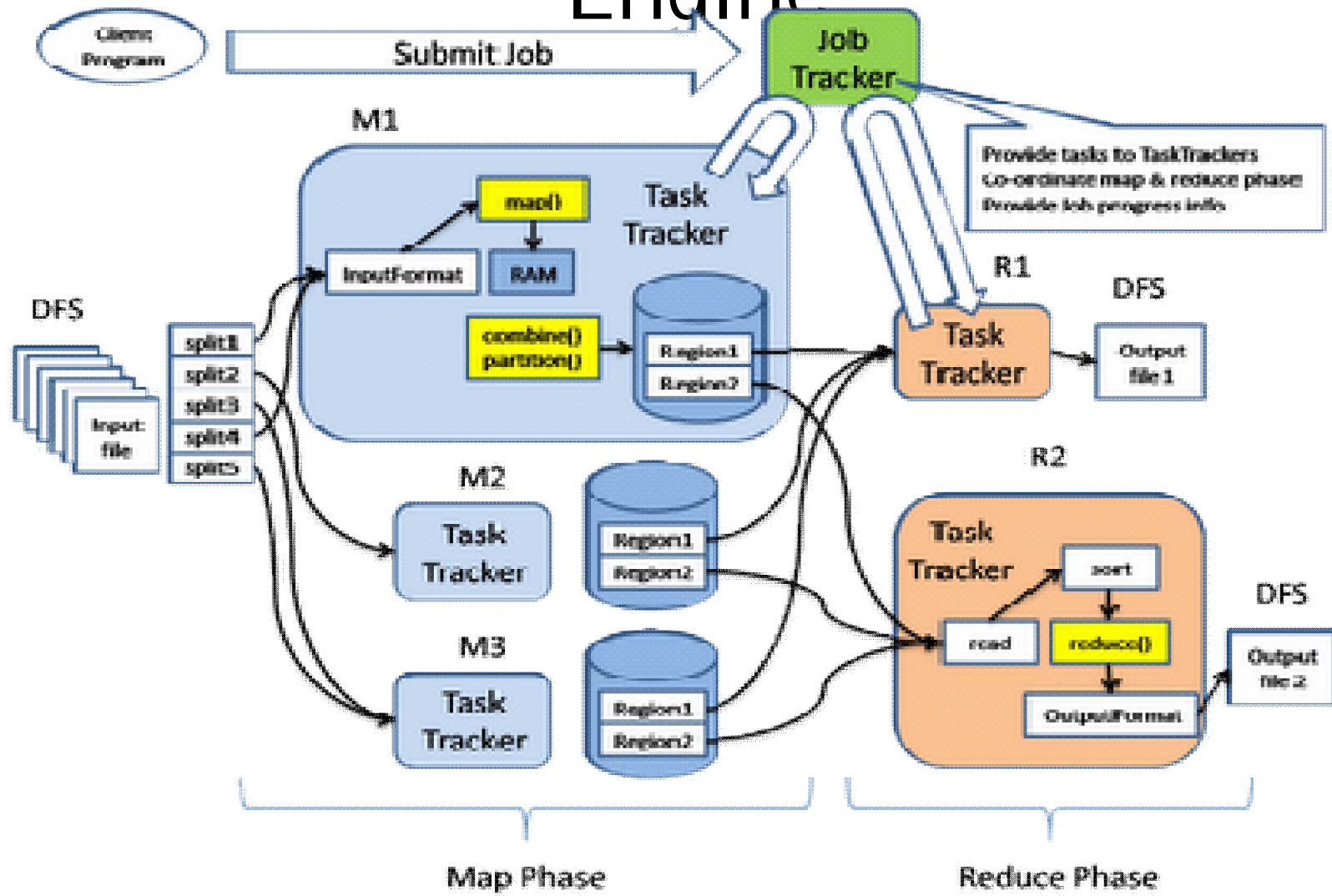
Hadoop's Developers

2005: Doug Cutting and Michael J. Cafarella developed Hadoop to support distribution for the [Nutch](#) search engine project.

The project was funded by Yahoo.

2006: Yahoo gave the project to Apache Software Foundation.

Hadoop's Architecture: MapReduce Engine



Hadoop Distributed FileSystem

- Tailored to needs of MapReduce
- Targeted towards many reads of filestreams
 - Writes are more costly
- Open Data Format
 - Flexible Schema
 - Queryable Database
- Fault Tolerance
 - High degree of data replication (3x by default)
 - No need for RAID on normal nodes
- Large blocksize (64MB)
- Location awareness of DataNodes in network

Hadoop Usage

- Non-realtime large dataset computing:
 - NY Times was dynamically generating PDFs of articles from 1851-1922
 - Wanted to pre-generate & statically serve articles to improve performance
 - Using Hadoop + MapReduce running on EC2 / S3, converted 4TB of TIFFs into 11 million PDF articles in 24 hrs

Hadoop Usage: Facebook Messages

- Design requirements:
 - Integrate display of email, SMS and chat messages between pairs and groups of users
 - Strong control over who users receive messages from
 - Suited for production use between 500 million people immediately after launch
 - Stringent latency & uptime requirements



Hadoop Usage: Facebook Messages

- System requirements
 - High write throughput
 - Cheap, elastic storage
 - Low latency
 - High consistency (within a single data center good enough)
 - Disk-efficient sequential and random read performance



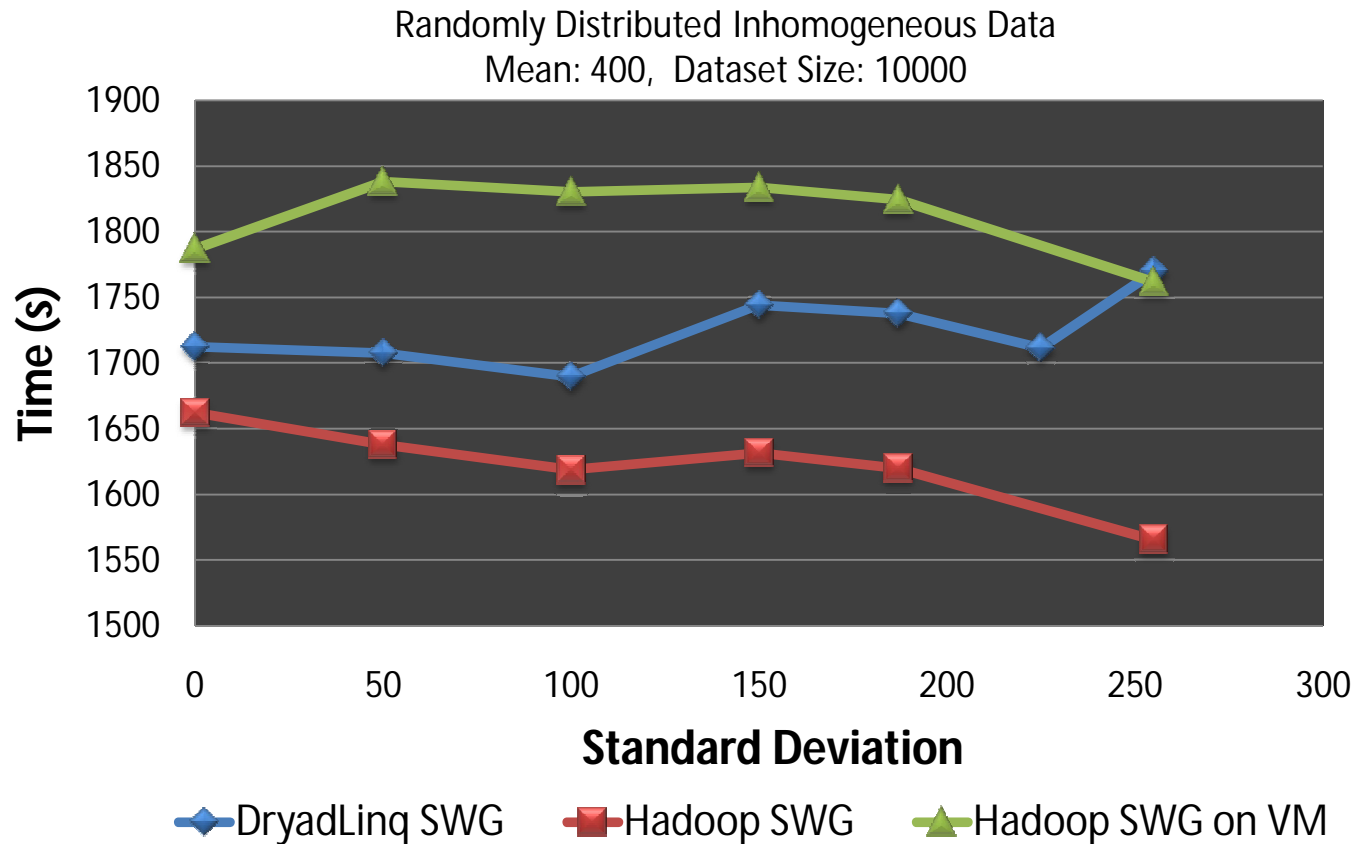
Hadoop Usage: Facebook Messages

- Classic alternatives
 - These requirements typically met using large MySQL cluster & caching tiers using Memcache
 - Content on HDFS could be loaded into MySQL or Memcached if needed by web tier
- Problems with previous solutions
 - MySQL has low random write throughput... BIG problem for messaging!
 - Difficult to scale MySQL clusters rapidly while maintaining performance
 - MySQL clusters have high management overhead, require more expensive hardware

Hadoop Usage: Facebook Messages

- Facebook's solution
 - Hadoop + HBase as foundations
 - Improve & adapt HDFS and HBase to scale to FB's workload and operational considerations
 - Major concern was availability: NameNode is SPOF & failover times are at least 20 minutes
 - Proprietary "AvatarNode": eliminates SPOF, makes HDFS safe to deploy even with 24/7 uptime requirement
 - Performance improvements for realtime workload: RPC timeout. Rather fail fast and try a different DataNode

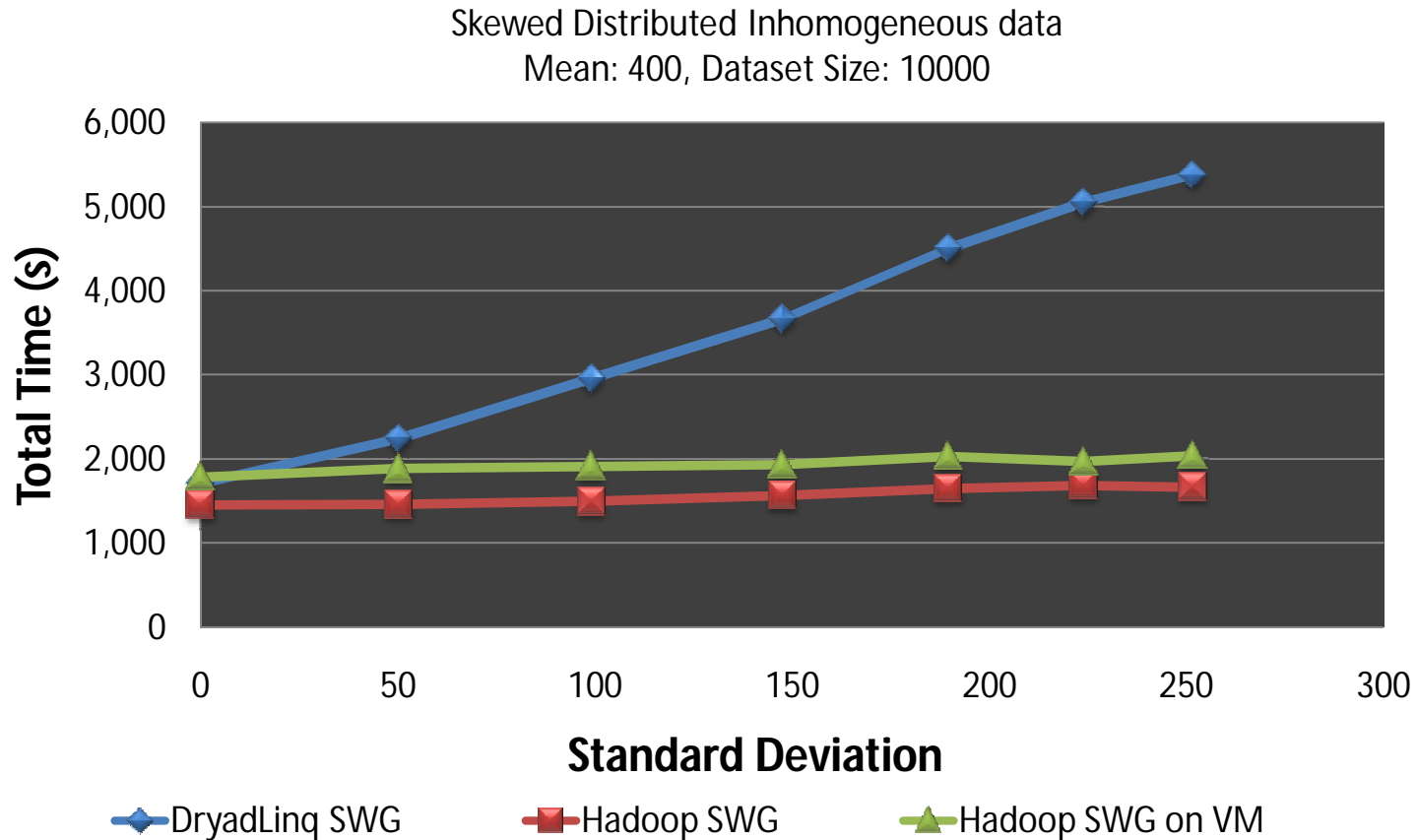
Inhomogeneous Data Performance



Inhomogeneity of data does not have a significant effect when the sequence lengths are randomly distributed

Dryad with Windows HPCS compared to Hadoop with Linux RHEL on Idataplex (32 nodes)

Inhomogeneous Data Performance



This shows the natural load balancing of Hadoop MR dynamic task assignment using a global pipe line in contrast to the DryadLinq static assignment
Dryad with Windows HPCS compared to Hadoop with Linux RHEL on Idataplex (32 nodes)

Pegasus and DAGMan

- Pegasus
 - Resource, data discovery
 - Mapping computation to resources
 - Orchestrate data transfers
 - Publish results
 - Graph optimizations
- DAGMAN
 - Submits tasks to execution resources
 - Monitor the execution
 - Retries in case of failure
 - Maintain dependencies

Some Other Applications of MapReduce

Distributed Grep:

- Input: large set of files
- Output: lines that match pattern

- Map – *Emits a line if it matches the supplied pattern*
- Reduce – *Copies the intermediate data to output*

Some Other Applications of MapReduce (2)

Reverse Web-Link Graph

- Input: Web graph: tuples (a, b) where (page a → page b)
- Output: For each page, list of pages that link *to* it

- Map – *process web log and for each input <source, target>, it outputs <target, source>*
- Reduce - *emits <target, list(source)>*

Some Other Applications of MapReduce (3)

Count of URL access frequency

- Input: Log of accessed URLs, e.g., from proxy server
- Output: For each URL, % of total accesses for that URL

- Map – *Process web log and outputs <URL, 1>*
- Multiple Reducers - *Emits <URL, URL_count>*
(So far, like Wordcount. But still need %)
- Chain another MapReduce job after above one
- Map – *Processes <URL, URL_count> and outputs <1, (<URL, URL_count>)>*
- 1 Reducer – Sums up *URL_count*'s to calculate *overall_count*.
Emits multiple <URL, URL_count/overall_count>

Some Other Applications of MapReduce (4)

Map task's output is sorted (e.g., quicksort)

Reduce task's input is sorted (e.g., mergesort)

Sort

- Input: Series of (key, value) pairs
- Output: Sorted <value>s

- Map – $\langle \text{key}, \text{value} \rangle \rightarrow \langle \text{value}, _ \rangle$ (*identity*)
- Reducer – $\langle \text{key}, \text{value} \rangle \rightarrow \langle \text{key}, \text{value} \rangle$ (*identity*)
- Partitioning function – partition keys across reducers based on ranges
 - Take data distribution into account to balance reducer tasks

UNIT V

Security Issues in the Cloud

- In theory, minimizing any of the issues would help:
 - Loss of Control
 - Take back control
 - Data and apps may still need to be on the cloud
 - But can they be managed in some way by the consumer?
 - Lack of trust
 - Increase trust (mechanisms)
 - Technology
 - Policy, regulation
 - Contracts (incentives): topic of a future talk
 - Multi-tenancy
 - Private cloud
 - Takes away the reasons to use a cloud in the first place
 - VPC: its still not a separate system
 - Strong separation

Minimize Lack of Trust: Certification

- Certification
 - Some form of reputable, independent, comparable assessment and description of security features and assurance
 - Sarbanes-Oxley, DIACAP, DISTCAP, etc (are they sufficient for a cloud environment?)
- Risk assessment
 - Performed by certified third parties
 - Provides consumers with additional assurance

Minimize Loss of Control in the Cloud

- Monitoring
- Utilizing different clouds
- Access control management

Minimize Loss of Control: Monitoring

- Cloud consumer needs situational awareness for critical applications
 - When underlying components fail, what is the effect of the failure to the mission logic
 - What recovery measures can be taken (by provider and consumer)
- Requires an application-specific run-time monitoring and management tool for the consumer
 - The cloud consumer and cloud provider have different views of the system
 - Enable both the provider and tenants to monitor the the components in the cloud that are under their control
 - Provide mechanisms that enable the provider to act on attacks he can handle.
 - infrastructure remapping (create new or move existing fault domains)
 - shutting down offending components or targets (and assisting tenants with porting if necessary)
 - Repairs
 - Provide mechanisms that enable the consumer to act on attacks that he can handle (application-level monitoring).
 - RAdAC (Risk-adaptable Access Control)
 - VM porting with remote attestation of target physical host
 - Provide ability to move the user's application to another cloud

Minimize Loss of Control: Utilize Different Clouds

- The concept of 'Don't put all your eggs in one basket'
 - Consumer may use services from different clouds through an intra-cloud or multi-cloud architecture
 - Propose a multi-cloud or intra-cloud architecture in which consumers
 - Spread the risk
 - Increase redundancy (per-task or per-application)
 - Increase chance of mission completion for critical applications
 - Possible issues to consider:
 - Policy incompatibility (combined, what is the overarching policy?)
 - Data dependency between clouds
 - Differing data semantics across clouds
 - Knowing when to utilize the redundancy feature (monitoring technology)
 - Is it worth it to spread your sensitive data across multiple clouds?
 - Redundancy could increase risk of exposure

Minimize Multi-tenancy in the Cloud

- Can't really force the provider to accept less tenants
 - Can try to increase isolation between tenants
 - Strong isolation techniques (VPC to some degree)
 - C.f. VM Side channel attacks (T. Ristenpart et al.)
 - QoS requirements need to be met
 - Policy specification
 - Can try to increase trust in the tenants
 - Who's the insider, where's the security boundary? Who can I trust?
 - Use SLAs to enforce trusted behavior

Taxonomy of Fear

- Confidentiality
 - Fear of loss of control over data
 - Will the sensitive data stored on a cloud remain confidential?
 - Will cloud compromises leak confidential client data
 - Will the cloud provider itself be honest and won't peek into the data?
- Integrity
 - How do I know that the cloud provider is doing the computations correctly?
 - How do I ensure that the cloud provider really stored my data without tampering with it?

Taxonomy of Fear (cont.)

- Availability
 - Will critical systems go down at the client, if the provider is attacked in a Denial of Service attack?
 - What happens if cloud provider goes out of business?
 - Would cloud scale well-enough?
 - Often-voiced concern
 - Although cloud providers argue their downtime compares well with cloud user's own data centers

Taxonomy of Fear (cont.)

- 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 information on clients
- Increased attack surface
 - Entity outside the organization now stores and computes data, and so
 - Attackers can now target the communication link between cloud provider and client
 - Cloud provider employees can be phished

Threat Model

- A threat model helps in analyzing a security problem, design mitigation strategies, and evaluate solutions
- Steps:
 - Identify attackers, assets, threats and other components
 - Rank the threats
 - Choose mitigation strategies
 - Build solutions based on the strategies

Threat Model

- Basic components
 - Attacker modeling
 - Choose what attacker to consider
 - insider vs. outsider?
 - single vs. collaborator?
 - Attacker motivation and capabilities
 - Attacker goals
 - Vulnerabilities / threats

What is the issue?

- The core issue here is the levels of trust
 - Many cloud computing providers trust their customers
 - Each customer is physically commingling its data with data from anybody else using the cloud while logically and virtually you have your own space
 - The way that the cloud provider implements security is typically focused on the fact that those outside of their cloud are evil, and those inside are good.
- But what if those inside are also evil?

Attacker Capability: Malicious Insiders

- At client
 - Learn passwords/authentication information
 - Gain control of the VMs
- At cloud provider
 - Log client communication
 - Can read unencrypted data
 - Can possibly peek into VMs, or make copies of VMs
 - Can monitor network communication, application patterns
 - Why?
 - Gain information about client data
 - Gain information on client behavior
 - Sell the information or use itself

Attacker Capability: Outside attacker

- What?
 - Listen to network traffic (passive)
 - Insert malicious traffic (active)
 - Probe cloud structure (active)
 - Launch DoS
- Goal?
 - Intrusion
 - Network analysis
 - Man in the middle
 - Cartography

The Network Level

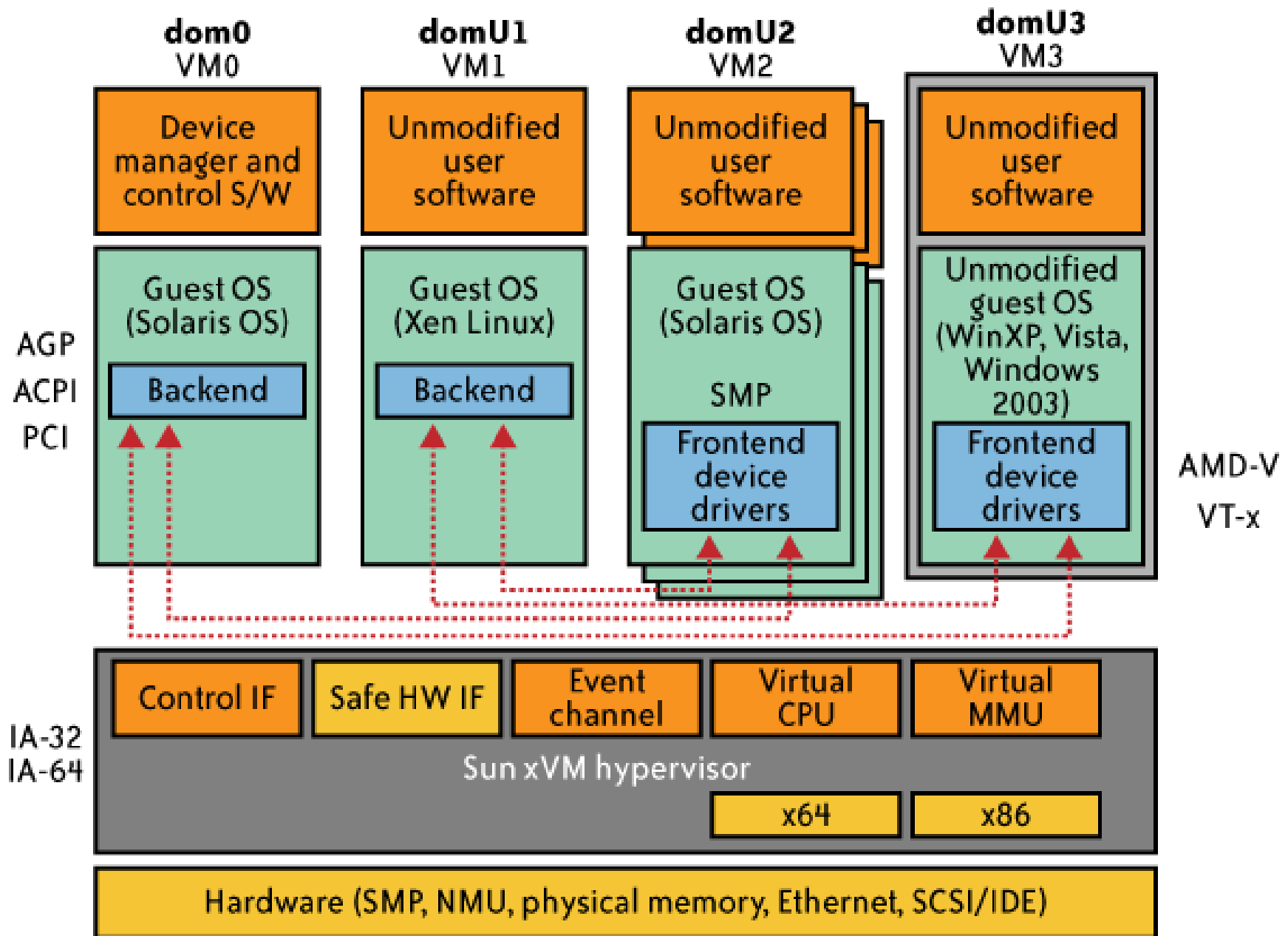
- Ensuring confidentiality and integrity of your organization's data-in-transit to and from your public cloud provider
- Ensuring proper access control (authentication, authorization, and auditing) to whatever resources you are using at your public cloud provider
- Ensuring availability of the Internet-facing resources in a public cloud that are being used by your organization, or have been assigned to your organization by your public cloud providers
- Replacing the established model of network zones and tiers with domains

The Network Level - Mitigation

- Note that network-level risks exist regardless of what aspects of “cloud computing” services are being used
- The primary determination of risk level is therefore not which *aaS is being used,
- But rather whether your organization intends to use or is using a public, private, or hybrid cloud.

The Host Level

- SaaS/PaaS
 - Both the PaaS and SaaS platforms abstract and hide the host OS from end users
 - Host security responsibilities are transferred to the CSP (Cloud Service Provider)
 - You do not have to worry about protecting hosts
 - However, as a customer, you still own the risk of managing information hosted in the cloud services.



Data Security and Storage

- Several aspects of data security, including:
 - Data-in-transit
 - Confidentiality + integrity using secured protocol
 - Confidentiality with non-secured protocol and encryption
 - Data-at-rest
 - Generally, not encrypted , since data is commingled with other users' data
 - Encryption if it is not associated with applications?
 - But how about indexing and searching?
 - Then homomorphic encryption vs. predicate encryption?
 - Processing of data, including multitenancy
 - For any application to process data, not encrypted

Data Security and Storage (cont.)

– Data lineage

- Knowing when an important for a cloud is
- e.g., Amazon AWS
 - Store <d1, t1, ex1.s3.amazonaws.com>
 - Process <d2, t2, ec2.compute2.amazonaws.com>
 - Restore <d3, t3, ex2.s3.amazonaws.com>

Where is (or was) that system located?
What was the state of that physical system?
How would a customer or auditor verify that info?

– Data provenance

- Computational accuracy (as well as data integrity)
- E.g., financial calculation: $\text{sum}(\frac{(((2 * 3) * 4)}{6}) - 2) = \$2.00 ?$
 - Correct : assuming US dollar
 - How about dollars of different countries?
 - Correct exchange rate?

Data Security and Storage

- Data remanence
 - Inadvertent disclosure of sensitive information is possible
- Data security mitigation?
 - Do not place any sensitive data in a public cloud
 - Encrypted data is placed into the cloud?
- Provider data and its security: storage
 - To the extent that quantities of data from many companies are centralized, this collection can become an attractive target for criminals
 - Moreover, the physical security of the data center and the trustworthiness of system administrators take on new importance.

What Are the Key Privacy Concerns?

- Typically mix security and privacy
- Some considerations to be aware of:
 - Storage
 - Retention
 - Destruction
 - Auditing, monitoring and risk management
 - Privacy breaches
 - Who is responsible for protecting privacy?

Storage

- Is it commingled with information from other organizations that use the same CSP?
- The aggregation of data raises new privacy issues
 - Some governments may decide to search through data without necessarily notifying the data owner, depending on where the data resides
- Whether the cloud provider itself has any right to see and access customer data?
- Some services today track user behaviour for a range of purposes, from sending targeted advertising to improving services

Retention

- How long is personal information (that is transferred to the cloud) retained?
- Which retention policy governs the data?
- Does the organization own the data, or the CSP?
- Who enforces the retention policy in the cloud, and how are exceptions to this policy (such as litigation holds) managed?

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?
- Cloud storage providers usually replicate the data across multiple systems and sites—increased availability is one of the benefits they provide.
 - How do you know that the CSP didn't retain additional copies?
 - Did the CSP really destroy the data, or just make it inaccessible to the organization?
 - Is the CSP keeping the information longer than necessary so that it can mine the data for its own use?

Auditing, monitoring and risk management

- How can organizations monitor their CSP and provide assurance to relevant stakeholders that privacy requirements are met when their PII is in the cloud?
- Are they regularly audited?
- What happens in the event of an incident?
- If business-critical processes are migrated to a cloud computing model, internal security processes need to evolve to allow multiple cloud providers to participate in those processes, as needed.
 - These include processes such as security monitoring, auditing, forensics, incident response, and business continuity

Privacy breaches

- How do you know that a breach has occurred?
- How do you ensure that the CSP notifies you when a breach occurs?
- Who is responsible for managing the breach notification process (and costs associated with the process)?
- If contracts include liability for breaches resulting from negligence of the CSP?
 - How is the contract enforced?
 - How is it determined who is at fault?