

Cloud Computing

Introduction

RIPON PATGIRI

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR

ASSAM, INDIA-788010

ripon@cse.nits.ac.in

ripon.patgiri@gmail.com

<http://cse.nits.ac.in/rp/>

Content I



Introduction

Syllabus & Text Book

Road map

Overview

Terminology

Taxonomy

Characteristics

On-demand self-service

Broad network access.

Measured service

Resource pooling

Rapid Elasticity

Service Models

Software as a Service

Platform as a Service

Infrastructure as a Service

Deployment Model

Content II

Private Cloud
Community Cloud
Public Cloud
Hybrid Cloud



Introduction

Syllabus & Text Book



CS 1482

Cloud Computing

3-0-0-6

8th sem (Open Elective)

Prerequisites: None

Introduction: Definition, Characteristics, Components, Cloud provider, SAAS, PAAS, IAAS and Others, Organizational scenarios of clouds, Administering & Monitoring cloud services, benefits and limitations, Deploy application over cloud, Comparison among SAAS, PAAS, IAAS

Cloud computing platforms: Infrastructure as service: Amazon EC2, Platform as Service: Google App Engine, Microsoft Azure, Utility Computing, Elastic Computing

Cloud Technologies: Study of Hypervisors, Compare SOAP and REST

Web services: SOAP and REST, SOAP versus REST, AJAX - asynchronous 'rich' interfaces, Mashups - user interface services

Virtualization: Virtual machine technology, virtualization applications in enterprises, Pitfalls of virtualization

Multitenant software: Multi-entity support, Multi-schema approach, Multi-tenancy using cloud data stores, Data access control for enterprise applications

Data in the cloud: Relational databases, Cloud file systems - GFS and HDFS, BigTable, HBase and Dynamo

Map-Reduce and extensions: Parallel computing, The map-Reduce model, Parallel efficiency of Map-Reduce, Relational operations using Map-Reduce, Enterprise batch processing using Map-Reduce, Introduction to cloud development, Example/Application of Mapreduce, Features and comparisons among GFS,HDFS etc, Map-Reduce model

Cloud security: Vulnerability assessment tool for cloud, Privacy and Security in cloud, Architectural Considerations - General Issues, Trusted Cloud computing, Secure Execution Environments and Communications, Security challenges - Virtualization security management- virtual threats, VM Security Recommendations, VM-Specific Security techniques, Secure Execution Environments and Communications in cloud

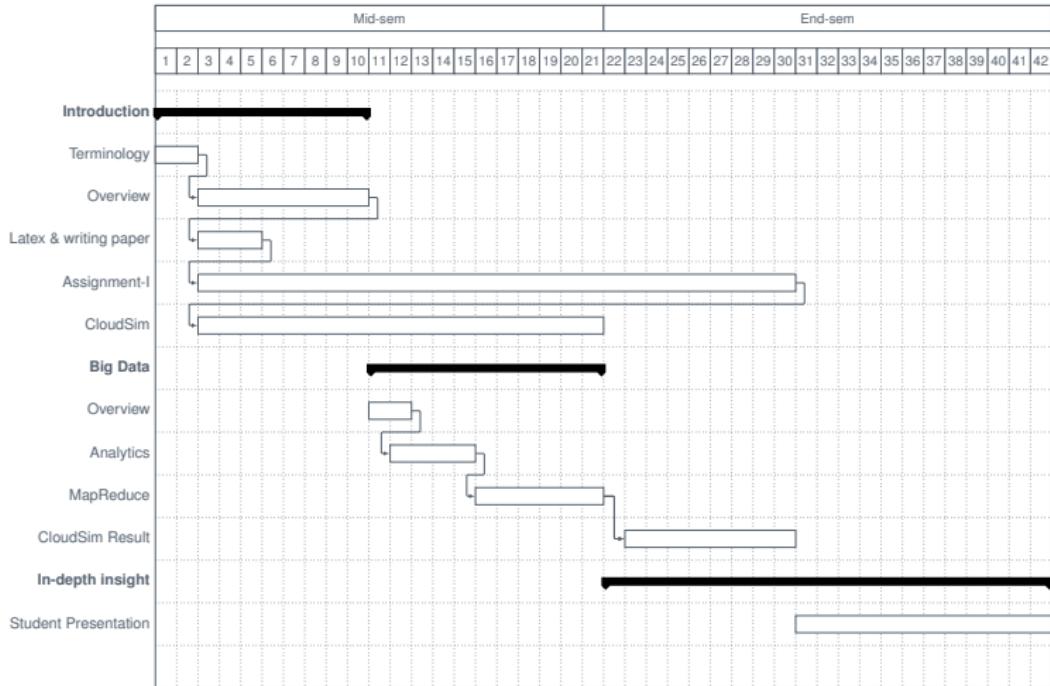
Issues: Implementing real time application over cloud platform
Issues in Intercloud environments, QOS Issues in Cloud, Dependability, data migration, streaming in Cloud. Quality of Service (QoS) monitoring in a Cloud computing environment

Books:

1. Cloud Computing for Dummies – Hurwitz J., Bloor R., Kanfman M., Halper F. (Wiley India)
2. Enterprise Cloud Computing – Shroff G. (Cambridge University Press)
3. Cloud Security – Krutz R., Vines R. D. (Wiley India)

Introduction

Road map





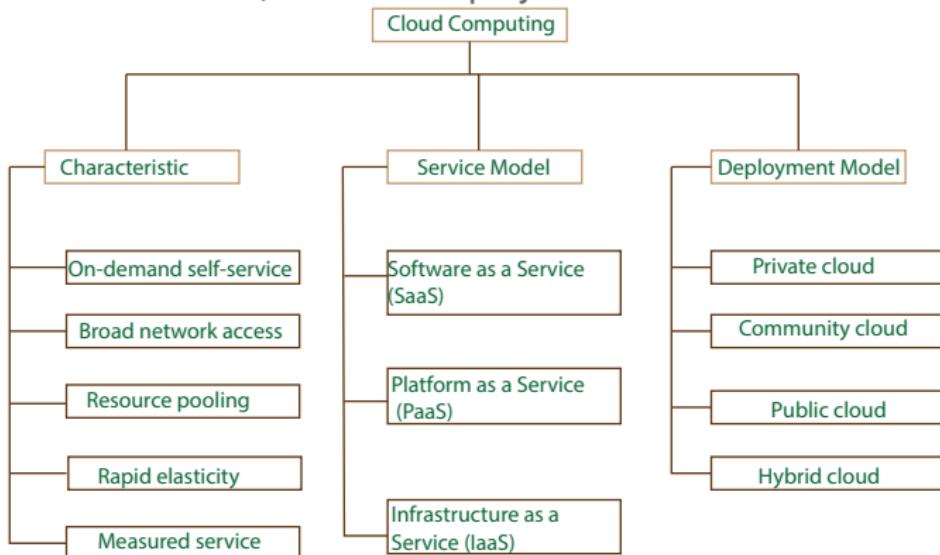
Definition

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.

Overview

Taxonomy

The cloud model is composed of five essential characteristics, three service models, and four deployment models.



Characteristics

On-demand self-service.



- ▶ A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider.

Characteristics

Broad network access.

- ▶ A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider.

Characteristics

Measured service



- ▶ Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts).
- ▶ Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service.

Characteristics

Resource pooling



10

This cloud model is composed of five essential characteristics, three service models, and four deployment models.

Characteristics

Resource pooling



This cloud model is composed of five essential characteristics, three service models, and four deployment models.

- ▶ The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand.

Characteristics

Resource pooling



10

This cloud model is composed of five essential characteristics, three service models, and four deployment models.

- ▶ The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand.
- ▶ There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or datacenter).
- ▶ Examples of resources include storage, processing, memory, and network bandwidth.

Characteristics

Rapid Elasticity



This cloud model is composed of five essential characteristics, three service models, and four deployment models.

- ▶ Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand.
- ▶ To the consumer, the capabilities available for provisioning often appear to be unlimited and can be appropriated in any quantity at any time.

Service Models

Software as a Service



- ▶ The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure.
- ▶ The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface.

Service Models

Software as a Service



- ▶ The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure.
- ▶ The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface.
- ▶ The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user specific application configuration settings.

Service Models

Platform as a Service

- ▶ The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider.

Service Models

Platform as a Service

- ▶ The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider.
- ▶ The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment.

Service Models

Infrastructure as a Service



- ▶ The capability provided to the consumer is to provision 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.



- ▶ The capability provided to the consumer is to provision 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.
- ▶ The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., host firewalls).

Deployment Model

Private Cloud



15

- ▶ The cloud infrastructure is provisioned for exclusive use by a single organization comprising multiple consumers (e.g., business units).
- ▶ It may be owned, managed, and operated by the organization, a third party, or some combination of them, and it may exist on or off premises.

Deployment Model

Community Cloud



- ▶ The cloud infrastructure is provisioned for exclusive use by a specific community of consumers from organizations that have shared concerns (e.g., mission, security requirements, policy, and compliance considerations).
- ▶ It may be owned, managed, and operated by one or more of the organizations in the community, a third party, or some combination of them, and it may exist **on or off premises**.

Deployment Model

Public Cloud



17

- ▶ The cloud infrastructure is provisioned for open use by the general public.
- ▶ It may be owned, managed, and operated by a business, academic, or government organization, or some combination of them.
- ▶ It exists on the premises of the cloud provider.

Deployment Model

Hybrid Cloud



18

- ▶ The cloud infrastructure is a composition of two or more distinct cloud infrastructures (private, community, or public) that remain unique entities, but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds).

Thank You

CLOUD COMPUTING

NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR

Course Code: CS-1482

Instructor:

RIPON PATGIRI
[Assistant Professor]



Outline

1 Introduction to Cloud Computing

- Benefits of Cloud computing

2 Challenges of Cloud computing

3 Issues of Cloud computing

4 Obstacles of Cloud Computing

5 Organization Goals for Primary Storage

6 Organization Goals for Primary Storage

Benefits of Cloud Computing

Benefits of Cloud Computing

Cost Reduce Cost. **Pay-as-you-go** model reduce the cost of clients. Moreover, cloud computing reduces the initial establishment of business.

Storage Increases Storage. The user need not to worry about their storage size. Users are given elastic storage spaces.

Benefits of Cloud Computing

Benefits of Cloud Computing

Cost Reduce Cost. **Pay-as-you-go** model reduce the cost of clients. Moreover, cloud computing reduces the initial establishment of business.

Storage Increases Storage. The user need not to worry about their storage size. Users are given elastic storage spaces.

Flexibility Better. The manageability of business, software, or anything in the cloud is much easier than traditional computing model. Due to cloud provider, the business become more flexible.

Disaster An user need not to worry about their data. The cloud provider always ensure data recovery in disaster.

Maintenance User need not to bear the cost of maintenance and hence, business become much easier.

Challenges of Cloud Computing

Challenges of Cloud Computing

Data Transfer Performance and data transfer rates become key issues as the distance between the data and the user increases.

Latency Even unlimited bandwidth without solving the latency problem will not improve the performance because it is the latency.

Challenges of Cloud Computing

Challenges of Cloud Computing

- Data Transfer** Performance and data transfer rates become key issues as the distance between the data and the user increases.
- Latency** Even unlimited bandwidth without solving the latency problem will not improve the performance because it is the latency.
- Bandwidth** Not all data access patterns are well suited to the cloud, particularly if there are large distances to cover. In such cases, bandwidth becomes not only a challenge but a financial consideration.
- Requirements** User requirements are unlimited bandwidth, high performance, low-response time, and mission-critical data.
- Storage** Maintenance of storage.

Issues of Cloud Computing

Issues of Cloud Computing

Security Security & Data Availability

Costs Costs.

Performance Performance

Latency Latency constraints

Management Manageability

Issues of Cloud Computing

Issues of Cloud Computing

Security Security & Data Availability

Costs Costs.

Performance Performance

Latency Latency constraints

Management Manageability

Interoperability A serious concern exists today is: Most of todays on-premises applications use block protocols such as FC, iSCSI etc. But Cloud storage protocols predominantly speak only in the language of file protocols (CIFS, NFS) and both public and private storage clouds are accessed via REST [] HTTP-based, or SOAP APIs. the figure ??, says that <http://cse.nits.ac.in/prasenjit/>

Contd.

Obstacles of Cloud Computing

There is always other side of a coin

Cloud Computing Adoption Obstacles

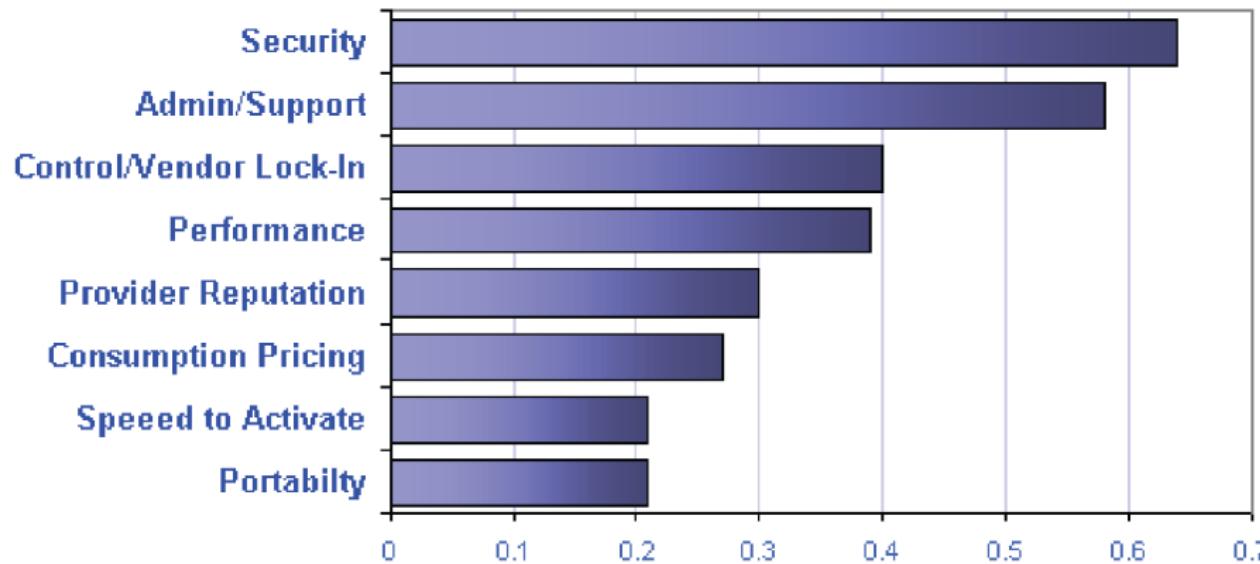


Figure: Obstacle of Cloud Computing

Organization Goals for Primary Storage

Organization Goals for Primary Storage

Economic Improved storage economics Leverage appropriate tiers of storage according to the performance requirement of the data.

Performance consistency working set and hotspot data to be automatically tiered to the highest-performance tier of storage (SSD), whereas non working-set or non hotspot data is tiered to a lower performing tier dynamically

Organization Goals for Primary Storage

Organization Goals for Primary Storage

Economic Improved storage economics Leverage appropriate tiers of storage according to the performance requirement of the data.

Performance consistency working set and hotspot data to be automatically tiered to the highest-performance tier of storage (SSD), whereas non working-set or non hotspot data is tiered to a lower performing tier dynamically

Transparent integration no complex policies to manage, no agents to install on servers, and the storage view from server to remain same in light of new integration with cloud storage

Deduplication - of primary storage to eliminate the repeated storage of redundant segments of data.

Organization Goals for Primary Storage

Organization Goals for Primary Storage

Economic Improved storage economics Leverage appropriate tiers of storage according to the performance requirement of the data.

Performance consistency working set and hotspot data to be automatically tiered to the highest-performance tier of storage (SSD), whereas non working-set or non hotspot data is tiered to a lower performing tier dynamically

Transparent integration no complex policies to manage, no agents to install on servers, and the storage view from server to remain same in light of new integration with cloud storage

Deduplication - of primary storage to eliminate the repeated storage of redundant segments of data.

Compression - Achieve higher levels of compression even when encountering single-byte insertion scenarios.

Data Reduction - Achieve 10X data reduction for specific storage workloads.

Organization Goals for Primary Storage

Organization Goals for Primary Storage

Version control support Version control can be enabled, thanks to some companies' implementation of primary storage de-duplication, which minimizes storage capacity requirements and cost

Pay-as-you-grow use of cloud storage enables pay-as-you-grow capacity consumption, which minimizes cost of over-provisioned yet unutilized on-premises storage

Organization Goals for Primary Storage

Organization Goals for Primary Storage

Version control support Version control can be enabled, thanks to some companies' implementation of primary storage de-duplication, which minimizes storage capacity requirements and cost

Pay-as-you-grow use of cloud storage enables pay-as-you-grow capacity consumption, which minimizes cost of over-provisioned yet unutilized on-premises storage

Optimal Cost Structure Using de-duplication, compression and encryption minimize cost for cloud storage capacity and cost for IO and data transfer,

Tape elimination Use Cloud Clones to enable consistent, point-in-time recovery snapshots and store independent copies in the cloud to eliminate need for tape



THANK YOU



INTERCONNECTED CLOUD COMPUTING ENVIRONMENTS-CHALLENGES

NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR

Course Code: CS-1482

Instructor:

RIPON PATGIRI
[Assistant Professor]



Outline

- 1 About the Paper
- 2 Provisioning
 - Discover
 - Selection
 - Allocation
- 3 Portability
 - VM Mobility
 - Data Portability
- 4 Security
- 5 Service-Level Agreement (SLA)
 - Trust
 - Authorization & Identity management
 - Federated SLA Management
 - Federated-Level Agreement
- 6 SLA Dependency
- 7 Legal Issues
- 8 Monitoring
- 9 Network
 - Connectivity
 - Addressing
 - Naming
 - Multicasting
- 10 Autonomy

Reference paper

Reference

Title: Interconnected Cloud Computing Environments:
Challenges, Taxonomy, and Survey

Author: ADEL NADJARAN TOOSI, RODRIGO N. CALHEIROS,
and RAJKUMAR BUYYA.

Affiliation: The University of Melbourne, Australia.

Journal: ACM Computing Survey, Volume 47, Issue 1, Article 7
(April 2014), 47 pages.

DOI: <http://dx.doi.org/10.1145/2593512>

Provisioning

Discover

- Cloud service discovery allows automatic detection of services and resources offered by cloud providers on the Internet. Since cloud providers offer a variety of services and use different ways to describe them, a way to provide a common access to cloud services and to discover and deploy them is necessary.

Provisioning

Discover

- Cloud service discovery allows automatic detection of services and resources offered by cloud providers on the Internet. Since cloud providers offer a variety of services and use different ways to describe them, a way to provide a common access to cloud services and to discover and deploy them is necessary.

Provisioning

Discover

- Cloud service discovery allows automatic detection of services and resources offered by cloud providers on the Internet. Since cloud providers offer a variety of services and use different ways to describe them, a way to provide a common access to cloud services and to discover and deploy them is necessary.
- Cloud customers require selection of the best possible application deployments in the cloud according to their objectives and constraints for QoS.

Provisioning

Discover

- Cloud service discovery allows automatic detection of services and resources offered by cloud providers on the Internet. Since cloud providers offer a variety of services and use different ways to describe them, a way to provide a common access to cloud services and to discover and deploy them is necessary.
- Cloud customers require selection of the best possible application deployments in the cloud according to their objectives and constraints for QoS.
- For instance, Google App Engine and Amazon EC2 do not offer discovery services, and Microsoft Azure and Force.com offer limited discovery.

Provisioning

Discover

- One of the main issues regarding service discovery in a multiple-cloud deployment is the lack of an integrated repository of cloud services.

Provisioning

Discover

- One of the main issues regarding service discovery in a multiple-cloud deployment is the lack of an integrated repository of cloud services.
- Another issue is that cloud providers describe their services with diverse languages, terms, and names. Moreover, there is not a common understanding regarding service functionalities, their QoS, and metrics among providers and customers.

Provisioning

Discover

- One of the main issues regarding service discovery in a multiple-cloud deployment is the lack of an integrated repository of cloud services.
- Another issue is that cloud providers describe their services with diverse languages, terms, and names. Moreover, there is not a common understanding regarding service functionalities, their QoS, and metrics among providers and customers.
- Finally, states of a large part of services in clouds change constantly and are dynamic in nature. The situation is even worse in interconnected cloud environments.

Provisioning

Selection

- Optimal application deployment in the cloud requires an effective selection strategy that works based on QoS criteria such as reliability, cost, and security and returns the set of the most suitable cloud services for end-customers.

Provisioning

Selection

- Optimal application deployment in the cloud requires an effective selection strategy that works based on QoS criteria such as reliability, cost, and security and returns the set of the most suitable cloud services for end-customers.
- Currently, selection is performed manually by cloud customers based on their requirements or through consultant companies.

Provisioning

Selection

- Optimal application deployment in the cloud requires an effective selection strategy that works based on QoS criteria such as reliability, cost, and security and returns the set of the most suitable cloud services for end-customers.
- Currently, selection is performed manually by cloud customers based on their requirements or through consultant companies.
- However, application deployment across multiple providers benefits from significant features such as the range of geographical locations, lower latency, higher reliability, lower deployment cost, higher failure resistance, and so forth.

Provisioning

Selection

- Optimal application deployment in the cloud requires an effective selection strategy that works based on QoS criteria such as reliability, cost, and security and returns the set of the most suitable cloud services for end-customers.
- Currently, selection is performed manually by cloud customers based on their requirements or through consultant companies.
- However, application deployment across multiple providers benefits from significant features such as the range of geographical locations, lower latency, higher reliability, lower deployment cost, higher failure resistance, and so forth.
- Consequently, an automated selection approach for application deployment is well motivated to optimize different aspects such as latency, reliability, throughput, data transfer, and cost.

Provisioning

Allocation

- Cloud providers usually offer their virtualized resources based on different QoS levels (e.g., best effort and reserved).

Provisioning

Allocation

- Cloud providers usually offer their virtualized resources based on different QoS levels (e.g., best effort and reserved).
- Physical resources in clouds are shared between cloud users.
Therefore, allocation strategies are needed to allocate resources to the requests in a profitable manner while fulfilling requests QoS requirements.

Provisioning

Allocation

- Cloud providers usually offer their virtualized resources based on different QoS levels (e.g., best effort and reserved).
- Physical resources in clouds are shared between cloud users. Therefore, allocation strategies are needed to allocate resources to the requests in a profitable manner while fulfilling requests QoS requirements.
- Contention happens when a user request cannot be admitted or cannot acquire sufficient resources because resources are occupied by other requests. This phenomenon is called resource contention.

Portability

VM Mobility

- VM Mobility is defined as the ability to move a running VM from one host to another without stopping it.

Portability

VM Mobility

- VM Mobility is defined as the ability to move a running VM from one host to another without stopping it.
- VM mobility requires- Memory and state transfer between hosts residing in different data centers.

Portability

VM Mobility

- VM Mobility is defined as the ability to move a running VM from one host to another without stopping it.
- VM mobility requires- Memory and state transfer between hosts residing in different data centers.
- VM mobility requires- Same LAN access by VMs at the destination host, without two sites sharing LAN.

Portability

VM Mobility

- VM Mobility is defined as the ability to move a running VM from one host to another without stopping it.
- VM mobility requires- Memory and state transfer between hosts residing in different data centers.
- VM mobility requires- Same LAN access by VMs at the destination host, without two sites sharing LAN.
- VM mobility requires- Same storage access by VMs at the destination host, without two sites sharing storage.

Portability

VM Mobility

- VM Mobility is defined as the ability to move a running VM from one host to another without stopping it.
- VM mobility requires- Memory and state transfer between hosts residing in different data centers.
- VM mobility requires- Same LAN access by VMs at the destination host, without two sites sharing LAN.
- VM mobility requires- Same storage access by VMs at the destination host, without two sites sharing storage.
- VM Mobility requires- a VM transfer from one physical machine to another without disrupting the network traffic flow.

Data Portability

Data Portability

- A cloud provider stores data in their own proprietary format, then users cannot move their data to other vendors without considerable cost and technical effort.

Data Portability

Data Portability

- A cloud provider stores data in their own proprietary format, then users cannot move their data to other vendors without considerable cost and technical effort.
- Therefore, industry standards and exporting tools, or at the very least formats that are publicly documented, are required to avoid data lock-in.

Data Portability

Data Portability

- A cloud provider stores data in their own proprietary format, then users cannot move their data to other vendors without considerable cost and technical effort.
- Therefore, industry standards and exporting tools, or at the very least formats that are publicly documented, are required to avoid data lock-in.
- Data portability is hindered by the lack of proper technology and standards and non-portability of the applications and data, which is exploited by cloud service providers for their own benefits

Data Portability

Avoid Vendor Lock-In

- Using APIs that have multiple independent implementations, for example, Amazon EC2 APIs, which are used by several others such as Eucalyptus.

Data Portability

Avoid Vendor Lock-In

- Using APIs that have multiple independent implementations, for example, Amazon EC2 APIs, which are used by several others such as Eucalyptus.
- Choosing a particular API that can run on multiple Clouds, for example, MapReduce and Hadoop.

Data Portability

Avoid Vendor Lock-In

- Using APIs that have multiple independent implementations, for example, Amazon EC2 APIs, which are used by several others such as Eucalyptus.
- Choosing a particular API that can run on multiple Clouds, for example, MapReduce and Hadoop.
- Manually decoupling the cloud-specific code of the application designed for each cloud provider from the application logic layer.

Data Portability

Avoid Vendor Lock-In

- Using APIs that have multiple independent implementations, for example, Amazon EC2 APIs, which are used by several others such as Eucalyptus.
- Choosing a particular API that can run on multiple Clouds, for example, MapReduce and Hadoop.
- Manually decoupling the cloud-specific code of the application designed for each cloud provider from the application logic layer.
- Creation of widespread standards and APIs.

Data Portability

Avoid Vendor Lock-In

- Using APIs that have multiple independent implementations, for example, Amazon EC2 APIs, which are used by several others such as Eucalyptus.
- Choosing a particular API that can run on multiple Clouds, for example, MapReduce and Hadoop.
- Manually decoupling the cloud-specific code of the application designed for each cloud provider from the application logic layer.
- Creation of widespread standards and APIs.
- Utilization of vendor-independent cloud abstraction layers such as jclouds and libcloud.

Security

Trust

- Trust typically refers to a situation where one party is willing to rely on the actions of another party.

- Trust typically refers to a situation where one party is willing to rely on the actions of another party.
- Ultimately, there is uncertainty as to whether the trusted party will behave or deliver as promised.

- Trust typically refers to a situation where one party is willing to rely on the actions of another party.
- Ultimately, there is uncertainty as to whether the trusted party will behave or deliver as promised.
- In the cloud computing environment, customers must trust in a cloud provider for the privacy and security of their assets (i.e., their data and processes).

- Trust typically refers to a situation where one party is willing to rely on the actions of another party.
- Ultimately, there is uncertainty as to whether the trusted party will behave or deliver as promised.
- In the cloud computing environment, customers must trust in a cloud provider for the privacy and security of their assets (i.e., their data and processes).
- In cloud computing, the risk of losing data confidentiality, integrity, and availability for customers is triggered by the lack of control over the data and processes.

- In the Inter-cloud scenario, the trust and reputation of a cloud provider affect other cloud providers.

- In the Inter-cloud scenario, the trust and reputation of a cloud provider affect other cloud providers.
- Trust federation is a combination of technology and policy infrastructure that allows organizations to trust each others verified users to enable the sharing of information, resources, and services in a secure and distributed way.

Security

Authorization & Identity management

- Identity Management (IdM) is an administrative task that deals with authentication of individuals in a system and authorization to access resources of the system based on the associated rights and restrictions.

Security

Authorization & Identity management

- Identity Management (IdM) is an administrative task that deals with authentication of individuals in a system and authorization to access resources of the system based on the associated rights and restrictions.
- In cloud computing environments, identity management services are mainly responsible for supporting access control to services based on user attributes (e.g., IP address, user and group name) and resource attributes (e.g., availability periods).

Security

Authorization & Identity management

- Identity Management (IdM) is an administrative task that deals with authentication of individuals in a system and authorization to access resources of the system based on the associated rights and restrictions.
- In cloud computing environments, identity management services are mainly responsible for supporting access control to services based on user attributes (e.g., IP address, user and group name) and resource attributes (e.g., availability periods).
- Identity management systems, in federated cloud environments, should allow identification of users and resources in addition to support interoperability across multiple identity domains.

Security

Authorization & Identity management

- Single Sign-On (SSO) authentication, where a cloud must be able to authenticate itself to gain access to the resources provided by federated foreign clouds belonging to the same trust context without further identity checks.

Security

Authorization & Identity management

- Single Sign-On (SSO) authentication, where a cloud must be able to authenticate itself to gain access to the resources provided by federated foreign clouds belonging to the same trust context without further identity checks.
- Digital identities and third parties, where a cloud has to be considered as a subject distinctively identified by credentials and each cloud must be able to authenticate itself with foreign clouds using its digital identity guaranteed by a third party.

- Cloud providers define (or negotiate with customers) a service-level agreement (SLA) to specify what they guarantee.

- Cloud providers define (or negotiate with customers) a service-level agreement (SLA) to specify what they guarantee.
- SLA is a contract that describes a service and, most importantly, sets the expected service-level objectives (QoS expectations).

- Cloud providers define (or negotiate with customers) a service-level agreement (SLA) to specify what they guarantee.
- SLA is a contract that describes a service and, most importantly, sets the expected service-level objectives (QoS expectations).
- Implementation of SLA mechanisms on top of federated resources is still an open question.

- In federated cloud environments, each participant cloud provider has its own SLA management mechanisms.

- In federated cloud environments, each participant cloud provider has its own SLA management mechanisms.
- The methods and protocols for negotiation of dynamic and flexible SLAs is required for dynamic environments such as Inter-cloud.

- In federated cloud environments, each participant cloud provider has its own SLA management mechanisms.
- The methods and protocols for negotiation of dynamic and flexible SLAs is required for dynamic environments such as Inter-cloud.
- An important issue in the federated cloud environment is how SLAs can be enforced in a federation where there are conflicting policies and goals of different members versus the objectives of the federation as a whole.

SLA

Federated-Level Agreement

- Federation-Level Agreement (FLA) that includes the set of rules and conditions that has to be signed by new providers once they join the federation.

- Federation-Level Agreement (FLA) that includes the set of rules and conditions that has to be signed by new providers once they join the federation.
- For example, a federation can set rules for minimum resources contributed to the federation or the set of QoS such as minimum expected availability.

- In federated cloud environments where a provider outsources its load to another provider, it expects a set of guaranteed QoS that is compatible to the promised QoS to end-users.

- In federated cloud environments where a provider outsources its load to another provider, it expects a set of guaranteed QoS that is compatible to the promised QoS to end-users.
- Quality of a service can be affected by external services.

- In federated cloud environments where a provider outsources its load to another provider, it expects a set of guaranteed QoS that is compatible to the promised QoS to end-users.
- Quality of a service can be affected by external services.
- It means that if one of the lower-layer services (e.g., infrastructure layer) is not functioning properly, it can affect the performance of higher-layer services (e.g., software layer).

- In federated cloud environments where a provider outsources its load to another provider, it expects a set of guaranteed QoS that is compatible to the promised QoS to end-users.
- Quality of a service can be affected by external services.
- It means that if one of the lower-layer services (e.g., infrastructure layer) is not functioning properly, it can affect the performance of higher-layer services (e.g., software layer).
- A practical approach is required to model the dependencies among services.

- Cloud computing by itself requires consideration of a broad set of legal issues such as privacy and security, contracting issues, issues related to location and ownership of data, and business considerations.

- Cloud computing by itself requires consideration of a broad set of legal issues such as privacy and security, contracting issues, issues related to location and ownership of data, and business considerations.
- When different organizations are involved in providing services for customers, one major issue is that it is difficult to guarantee confidentiality and privacy on data, especially when data is located in different countries with different laws.

- Cloud computing by itself requires consideration of a broad set of legal issues such as privacy and security, contracting issues, issues related to location and ownership of data, and business considerations.
- When different organizations are involved in providing services for customers, one major issue is that it is difficult to guarantee confidentiality and privacy on data, especially when data is located in different countries with different laws.
- For example, in a federated scenario in which a cloud provider leverages another providers services and customers might not generally have control of where the service they are leasing is operating, in case of failures in the service delivery it will be difficult for the cloud user to identify the real causes.

Monitoring

Monitoring

Monitoring

Example

Connectivity

Connectivity

Connectivity

Example

Addressing

Addressing

Addressing

Example

Naming

Naming

Naming

Example

Multicasting

Multicasting

Multicasting

Example

Autonomy

Autonomy

Autonomy

Example

THANK YOU

HADOOP

DR. RIPON PATGIRI

NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR

MAPREDUCE

- The MapReduce programming paradigm is the best parallel programming concept even if the MapReduce is purely based on the only Map and Reduce task.
- MapReduce can solve almost every problem of distributed and parallel computing, and large-scale data-intensive computing.
- The MapReduce programming works on low-cost unreliable commodity hardware, and it is an extremely scalable RAIN cluster, fault tolerant yet easy to administer, and highly parallel yet abstracted.
- The MapReduce is key/value pair computing wherein the input and output are in the form of key and value.

MAPPER

- The Map function transforms the input key/value pair to intermediate key-value pair. For instance, the key is the file name and value is its content.
- The output of Map function is transferred to reducer function by shuffling and the sorting.
- The input is fetched from the file system, namely, HDFS, and GFS.

REDUCER

- The Reduce function consists of three different phases, namely, copy phase, shuffle phase and reduce phase.
- The copy phase fetches the output from the Map function. The Map function spills the output when it reaches a specific size (for example, 10MB).
- This spilling cause early starting of Reduce task, otherwise the reduce task has to wait until the Map task has not finished.
- The shuffle phase sort the data using an external sorting algorithm. The shuffle phase and copy phase takes a longer time to execute as compared other phases.
- The reduce phase computes as the user defines and produces final output. The final output is written to the file system.

MAPREDUCE

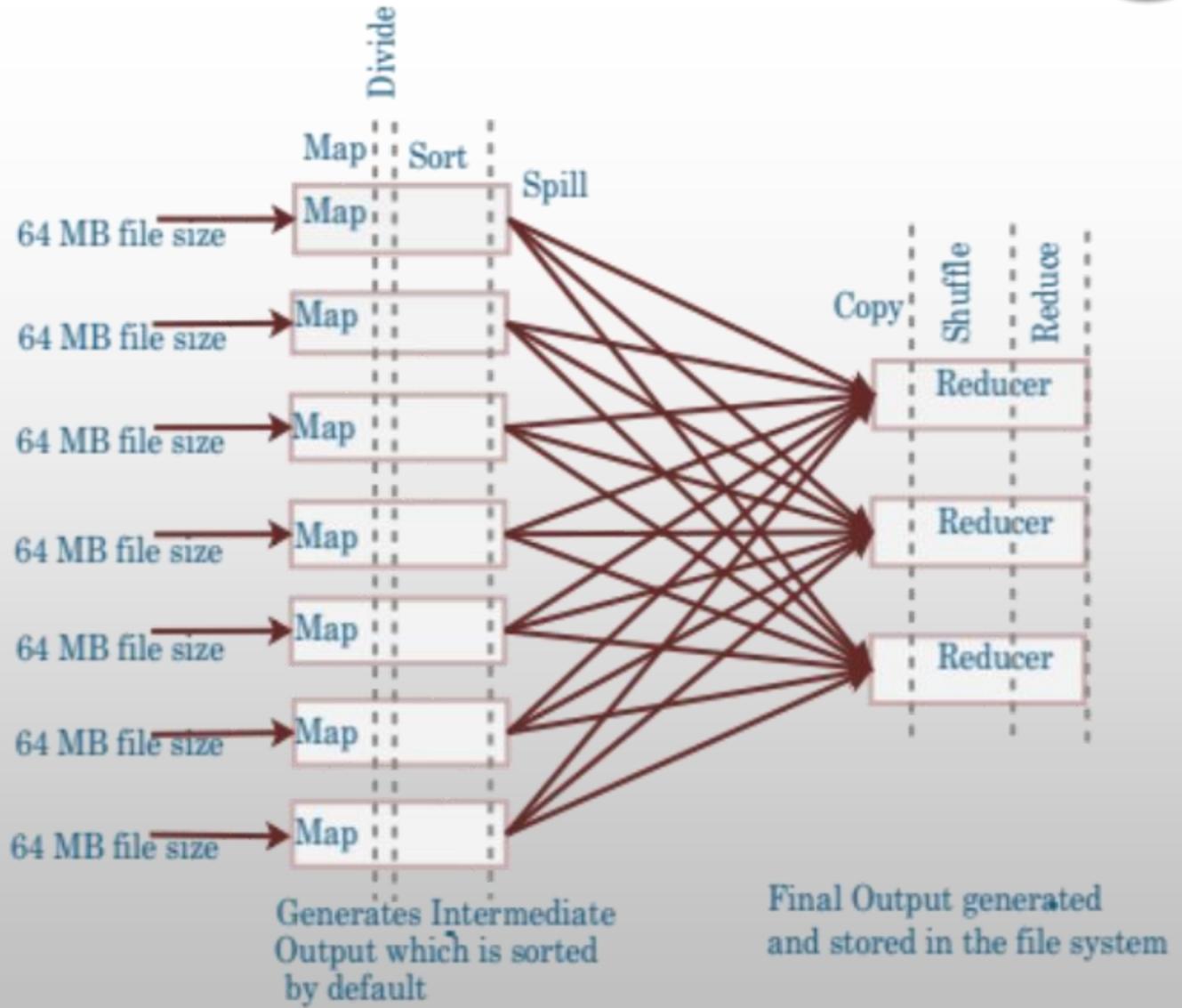
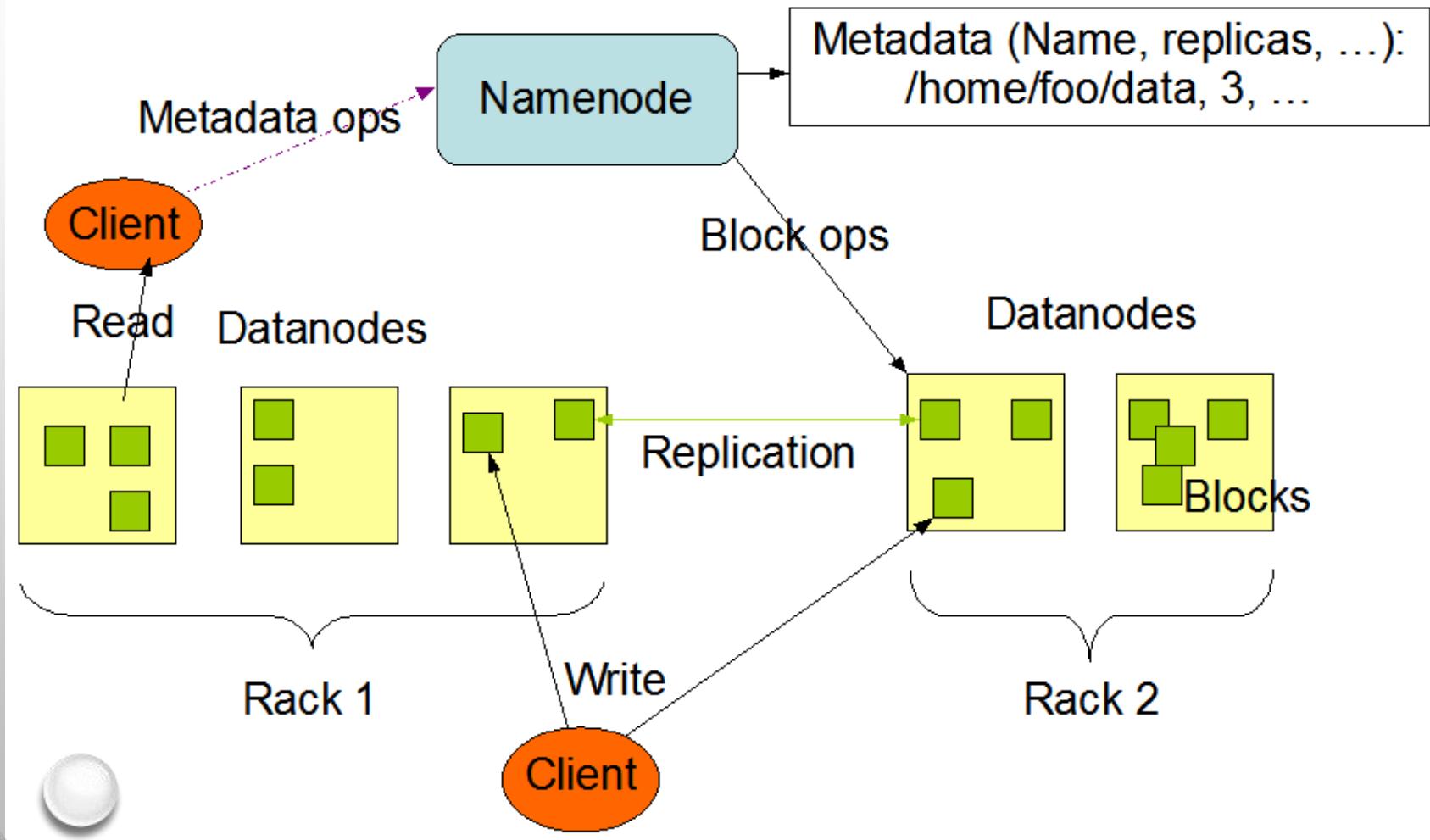


Fig. 4: Architecture of MapReduce

HDFS

HDFS Architecture



ASSUMPTION

- HARDWARE FAILURE
- LARGE DATA SETS
- SIMPLE COHERENCY MODEL
- MOVING COMPUTATION IS CHEAPER THAN MOVING DATA
- PORTABILITY ACROSS HETEROGENEOUS HARDWARE AND SOFTWARE PLATFORMS

NAMENODE

- Master node- It is also known as metadata server.
- It maintains the metadata of each blocks of data stored in DataNode.
- The NameNode executes file system namespace operations like opening, closing, and renaming files and directories.
- It also determines the mapping of blocks to DataNodes.
- The DataNodes are responsible for serving read and write requests from the file system's clients.
- The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.

NAMENODE

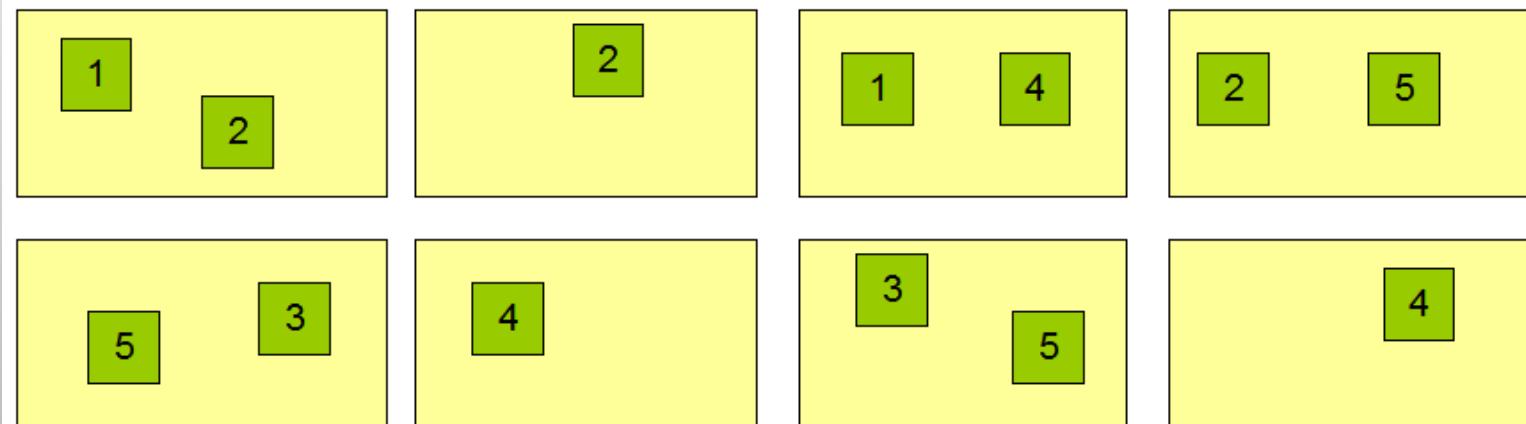
- Secondary NameNode is backup node of NameNode.
- It is not hot-standby node. It is used to restart the NameNode, if fails.
- Failure of NameNode cause single point of failure (SPoF).
- To overcome SPoF, AvatarNode is created by Facebook inc.

DATA REPLICATION

Block Replication

Namenode (Filename, numReplicas, block-ids, ...)
/users/sameerp/data/part-0, r:2, {1,3}, ...
/users/sameerp/data/part-1, r:3, {2,4,5}, ...

Datanodes



DR. HADOOP

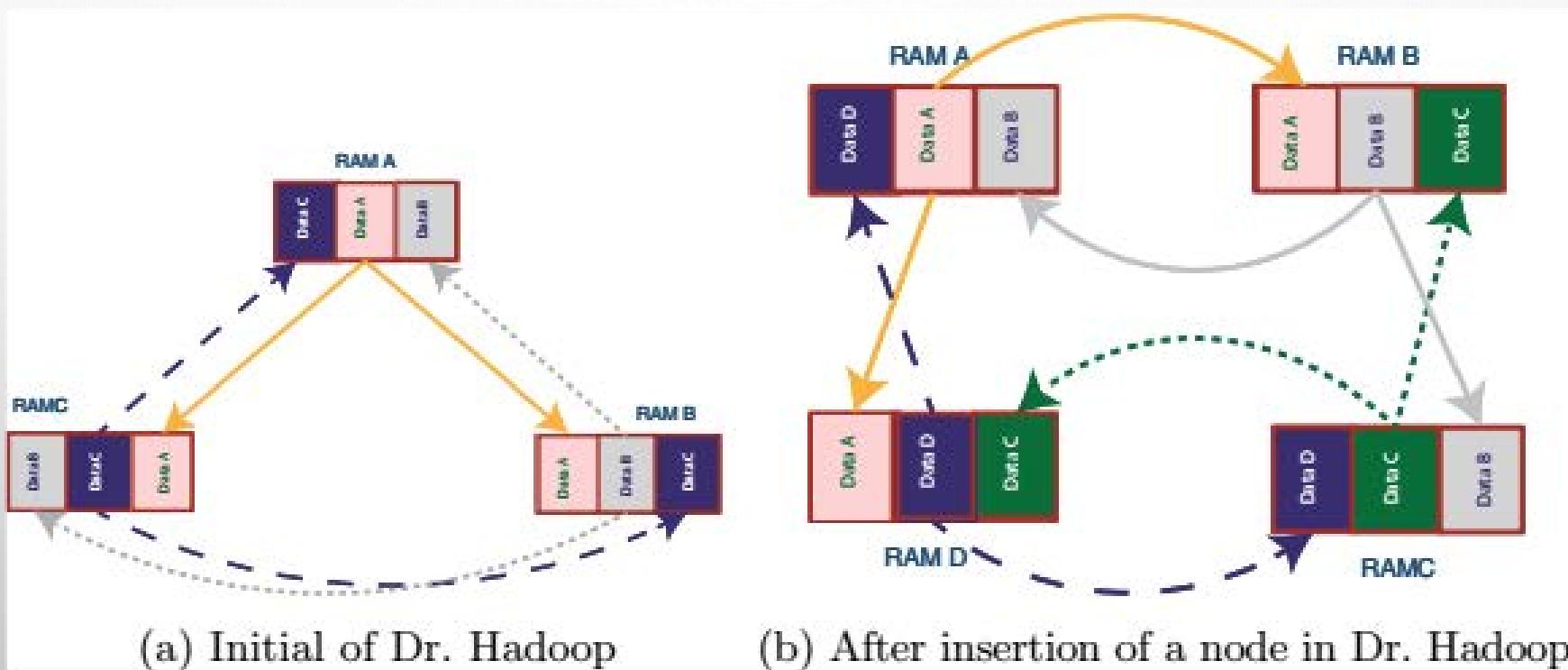


Fig. 1 Dr. Hadoop initialization and four nodes of Dr. Hadoop

DR. HADOOP

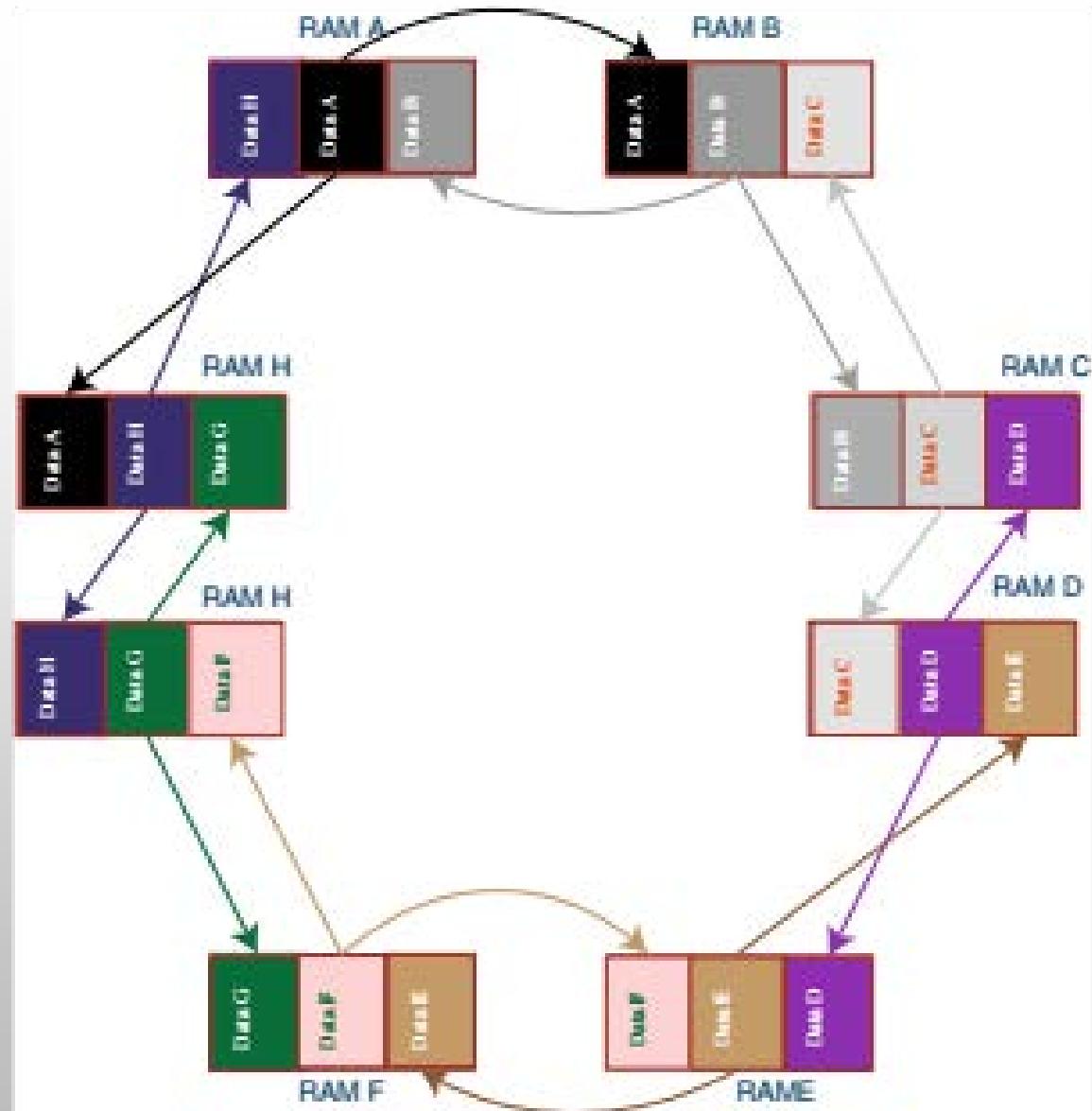


Fig. 3 In-memory replication design

DR. HADOOP

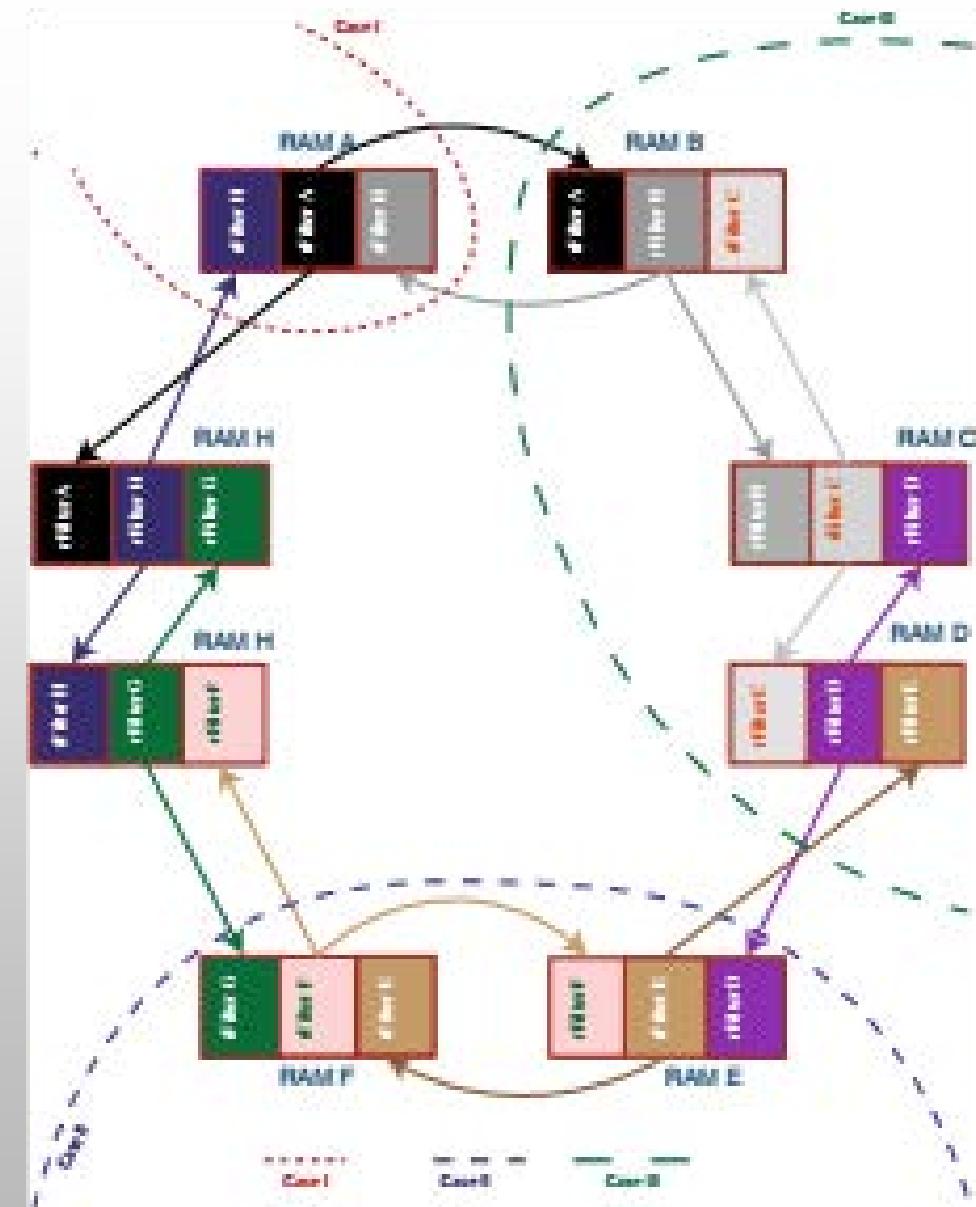


Fig. 4 Cases of failures in Dr. Hadoop

STORAGE SYSTEMS

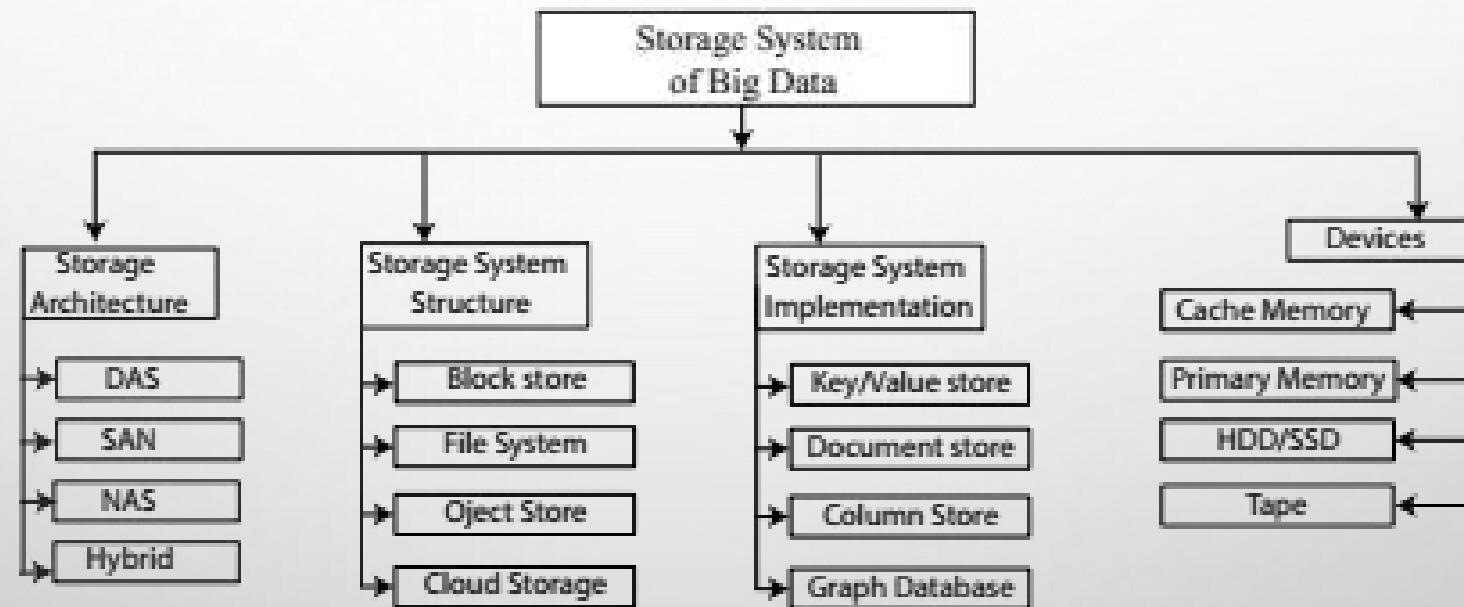


Fig. 5: Storage System

STORAGE SYSTEMS

TABLE 2: Storage Architecture comparison of features

Features	DAS	NAS	SAN
Data Transmission	IDE/SCSI	TCP/IP	Fibre
Storage Type	Track & Sector	Shared Files	Blocks
Fault Tolerance	RAID	Replication	RAID
Scalability	Not Scalable	Scalable	Limited
Distance coverage	Within a system	Very Long Distance	Very Short Distance
Pros	Very simple architecture, easy to manage, ideal for local services	Unbound scalability, distance does not matter	Very fast access of storage device
Cons	Not scalable	Not fast as well as SAN	Complex scalable, distance coverage is a problem.

STORAGE SYSTEMS

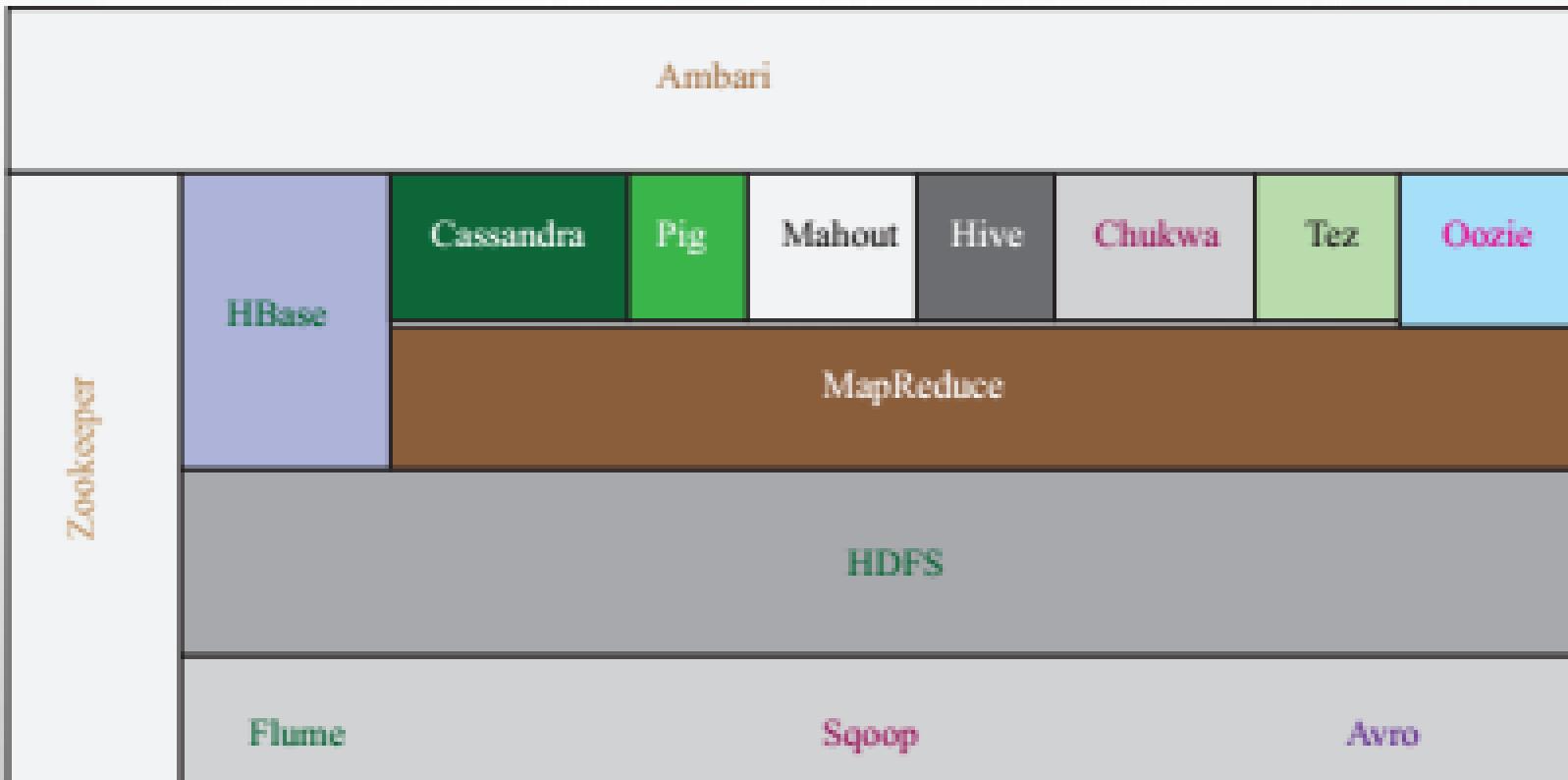
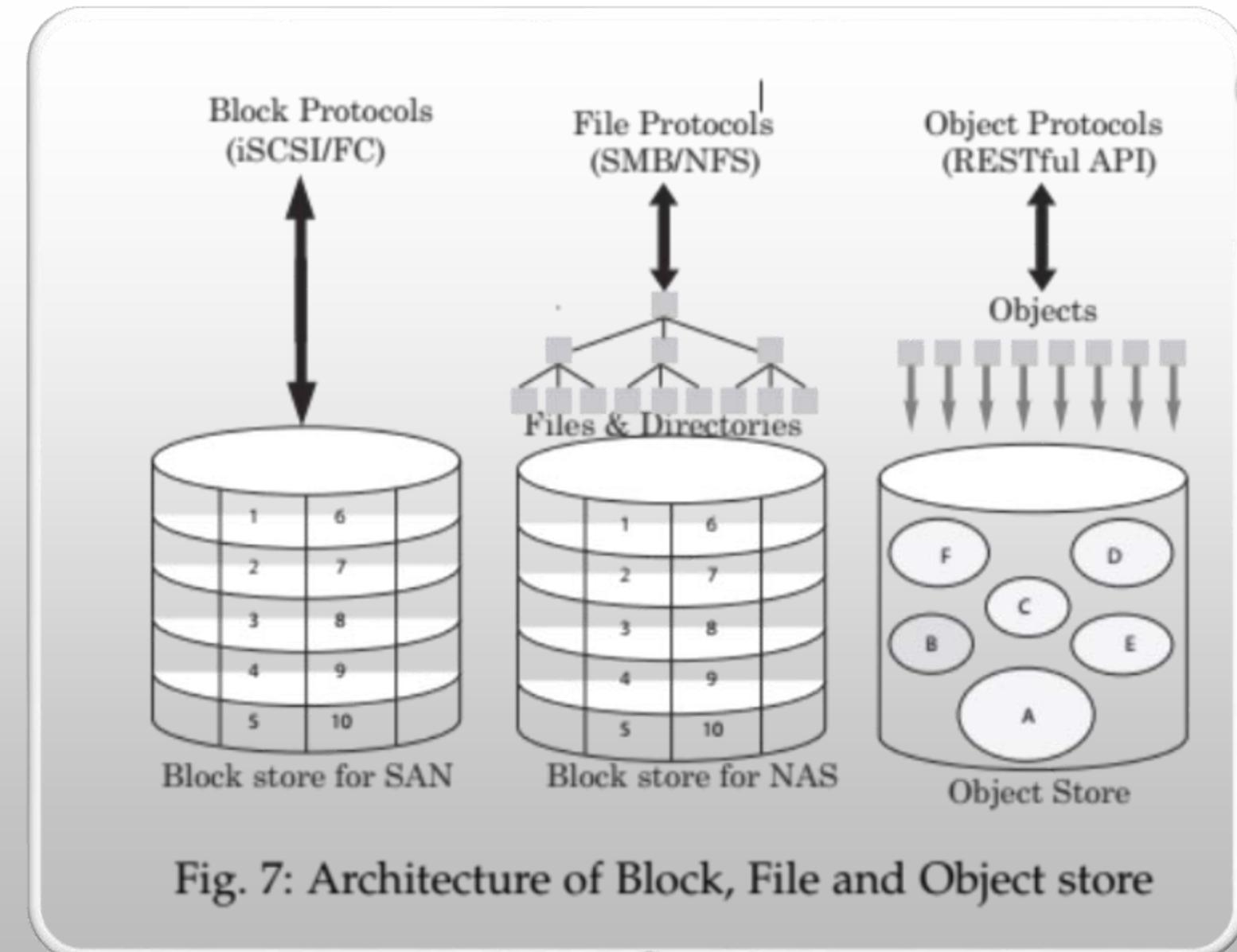


Fig. 6: The Hadoop Stack

STORAGE SYSTEMS



STORAGE SYSTEMS

TABLE 3: Modern File System comparison, * denotes limited, × denotes no and ✓ denotes yes

Name	Scalability	Disaster Recovery	Hot-standby	SPoF problem	POSIX-compliant	MDS
HDFS [8]	*	×	×	✓	×	Standalone
GFS [9]	*	×	×	✓	×	Standalone
CephFS [46]	✓	×	✓	✗	✓	Distributed
QFS [50]	*	×	✗	✓	✗	Standalone
BatchFS [51]	✓	×	✓	✗	✓	Distributed
Dr. Hadoop [41]	✓	×	✓	✗	✗	Distributed
ShardFS [45]	✓	×	✓		✓	Distributed
DMooseFS [52]	✓	×	✓	✗	✗	Distributed
CalvinFS [42]	✓	✓	✓	✗	✓	Distributed
GPPS [53]	✓	×	✓	✓	✓	Parallel
GlusterFS [54]	✓	✓	✓	✗	✓	FUSE
DeltaFS [55]	✓	×	✓	✗	✗	LevelDB

**18th IEEE International Conference on High Performance Computing
and Communications (HPCC 2016)**
Sydney, Australia

Big Data

The V's of the Game Changer Paradigm

¹RIPON PATGIRI, AND ARIF AHMED

NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR
ASSAM, INDIA

{ripon,arif}@cse.nits.ac.in

¹<http://cse.nits.ac.in/rp/>

12-14, December, 2016



Introduction

Data, and Big Data

Motivation

Big Data

Facts on Big Data

V's of Big Data

Doug Laney's V

Other V's of Big Data

Confusion

New V's to the Big Data World

Volume redefinition

Two new V's

Total V's of Big Data

Total V's of Big Data

Conclusion

Introduction

Data, and Big Data

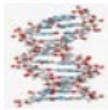


- ▶ The Big Data is large set of data to store, process and manage.
- ▶ The Big Data is defined as Game Changer Paradigm.
- ▶ It has nothing untouched area where data matters.

Introduction

Data, and Big Data

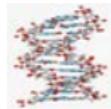
- ▶ The Big Data is large set of data to store, process and manage.
- ▶ The Big Data is defined as Game Changer Paradigm.
- ▶ It has nothing untouched area where data matters.



Introduction

Data, and Big Data

- ▶ The Big Data is large set of data to store, process and manage.
- ▶ The Big Data is defined as Game Changer Paradigm.
- ▶ It has nothing untouched area where data matters.



Introduction

Motivation



Controversy: How many V's are there in Big Data paradigm?

Information: What are the widely accepted V's of Big Data?

Confusion: What are the correct V's of Big Data?

Introduction

Motivation



Controversy: How many V's are there in Big Data paradigm?

Information: What are the widely accepted V's of Big Data?

Confusion: What are the correct V's of Big Data?

All V's of Big Data-Volume \neq Big Data.

Introduction

Motivation



Controversy: How many V's are there in Big Data paradigm?

Information: What are the widely accepted V's of Big Data?

Confusion: What are the correct V's of Big Data?

All V's of Big Data-Volume \neq Big Data.

Contribution

V_3 : Defining the volume of Big Data.

Coining: Coining two V's.

Big Data

Facts on Big Data



Figure: The most famous infographic of Big Data, by IDC.com & EMC²

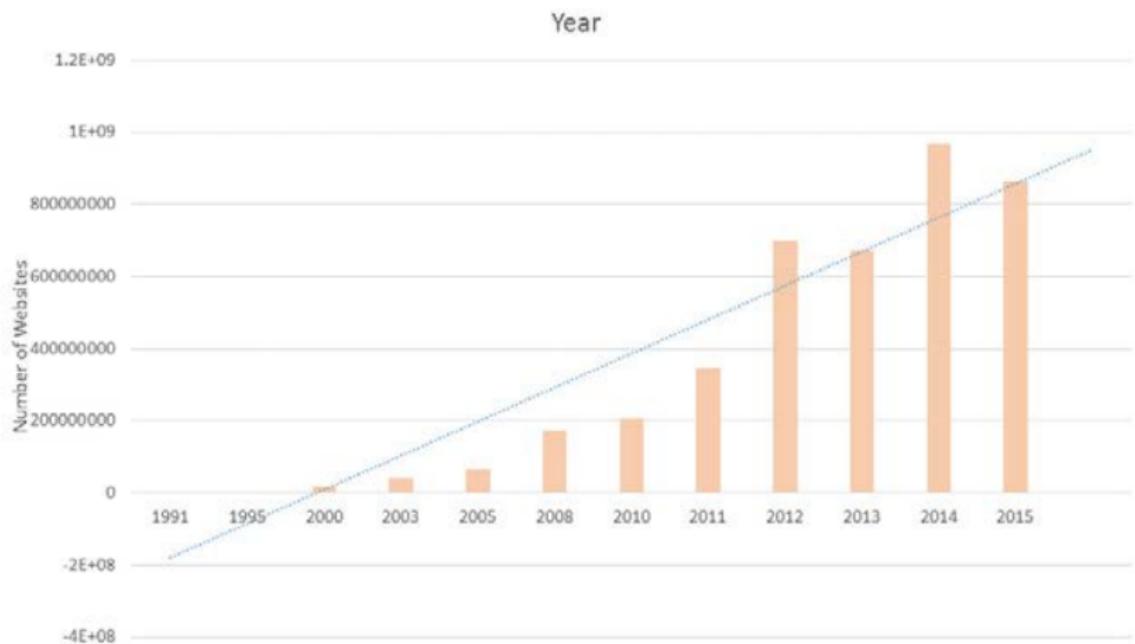


Figure: Growth of website per year [8]

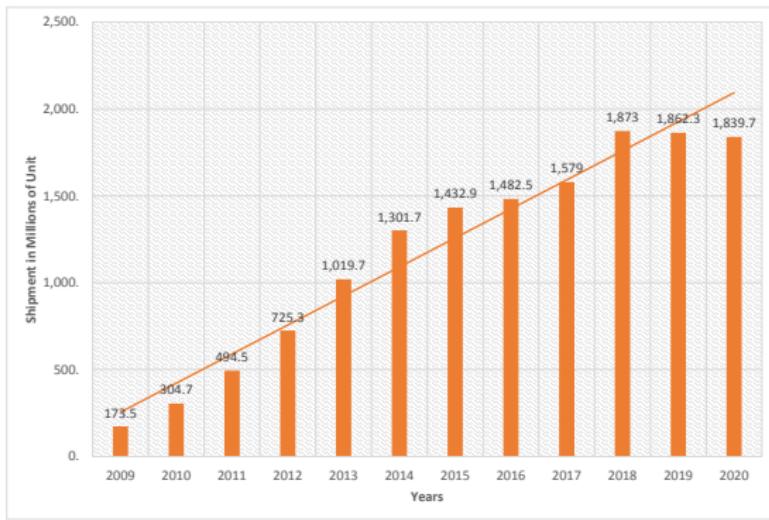


Figure: Shipment of smartphone [9]

Big Data

Facts on Big Data

7

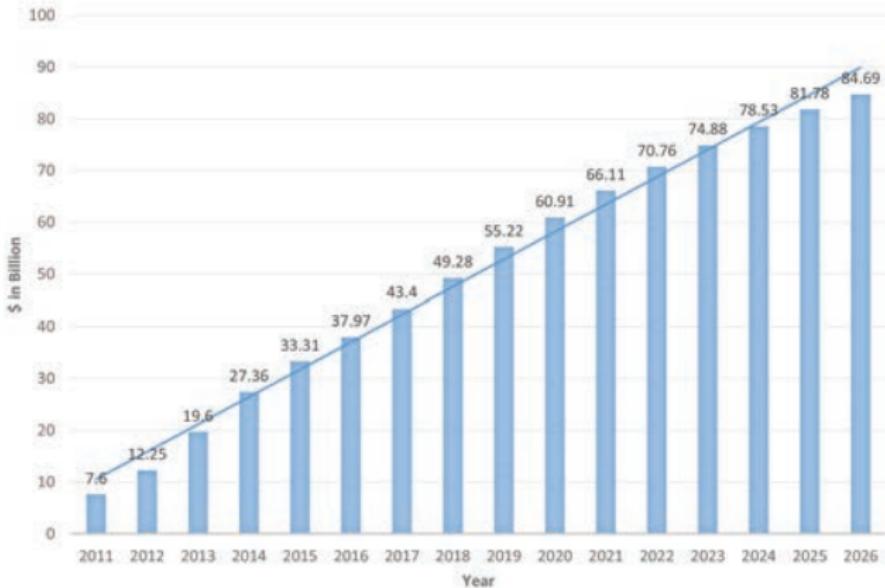


Figure: Big Data revenue in USD [10]

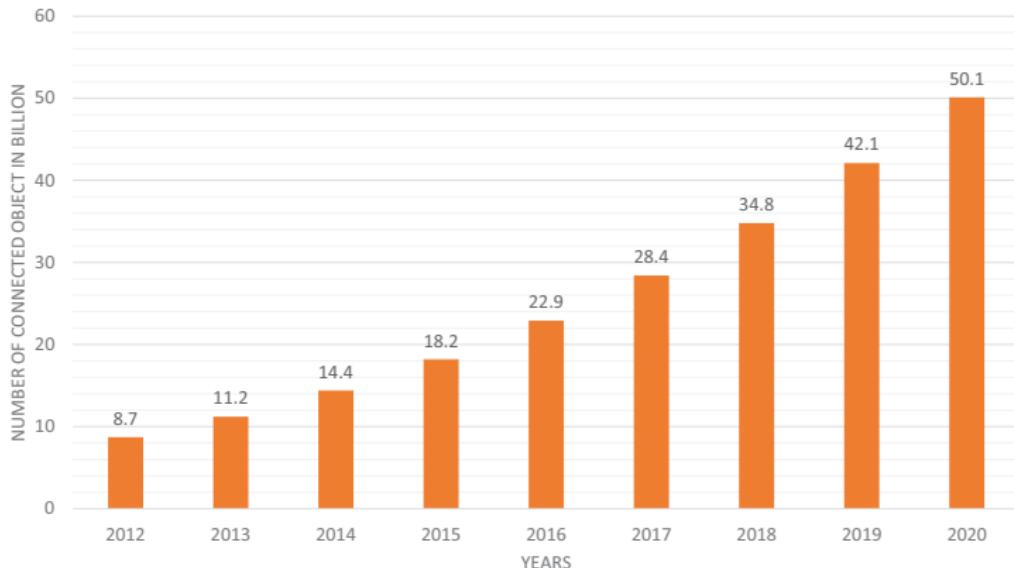


Figure: Growth of connected object [11]

V's of Big Data

Doug Laney's V

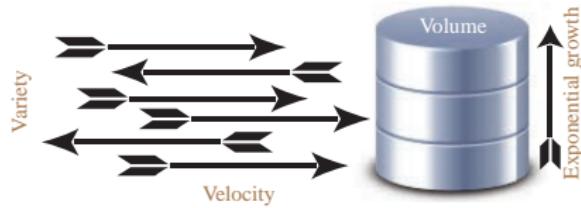


Figure: Relationship among Volume, Velocity, & Variety

V's of Big Data

Doug Laney's V

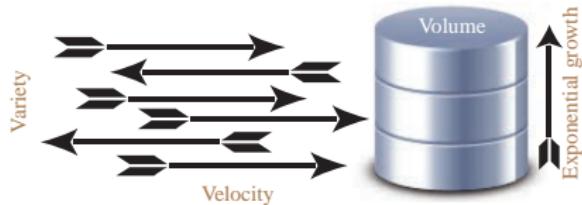


Figure: Relationship among Volume, Velocity, & Variety

Volume: Petabytes, Exabytes, Zettabytes, and Yottabytes [1].

Velocity: Growth and Transmission [1].

Variety: Structured, unstructured, and semi-structured [1].

Other V's of Big Data

Veracity, Validity, and Value



Veracity: Question: Can we believe an operation on very large-set of data? Trustworthy, Meaningful, Accuracy [2].



Validity: $Volume - Validity = Worthlessness$ [2].

Value: $BigData = Volume + Value$ [2].

Other V's of Big Data

Virtual, and Complexity

Virtual: Data Management is virtual [7]. Actual representation and virtual differs.

Complexity: Complexity is integral part of computation [4]. The high complexity is always associated with Big Data.

Other V's of Big Data

Confusion



12

Visibility: The visibility is the state of being able to see or be seen [6].

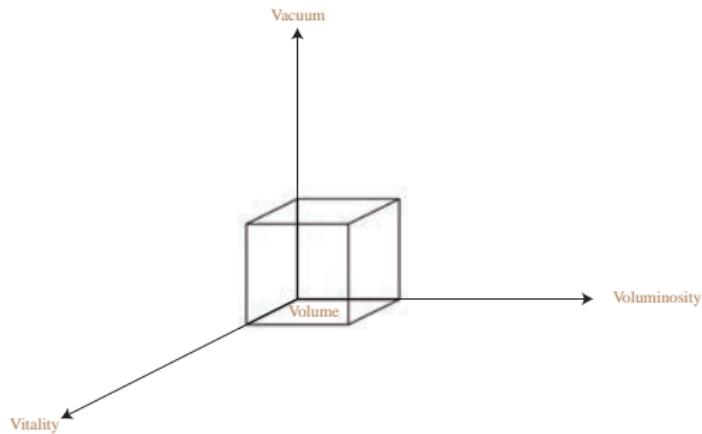
✓ **Visualization:** Show the hidden data. Stitching the data to show or make visualization [3].

✓ **Variability:** Modification, mutation, changing, and shifting intentionally [5].

Volatility: Change abruptly, sudden change, instability, and changed by anonymously or unintentionally [2].

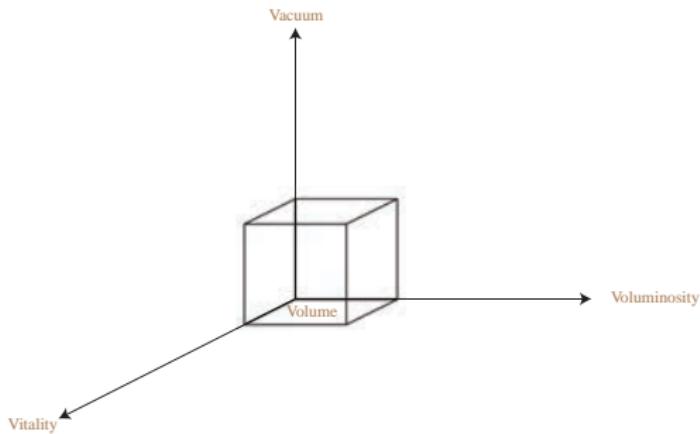
New V's to the Big Data World

Volume redefinition



New V's to the Big Data World

Volume redefinition



Voluminosity: Greatness in Volume. Large set of Data.

Vacuum: Empty space to store large set of data. Creating empty space to store more incoming data.

Vitality: The power of enduring, survive and grow. The vitality refers survival of large-set of data, growth and actively served data.

New V's to the Big Data World

Two new V's



New V's to the Big Data World

Two new V's



Vendee: Massive users. For instance, massive users in banking.

Vase: Foundation of Big Data. Infrastructure to deal with large-scale of data.

Total V's of Big Data

All V's of Big Data

15

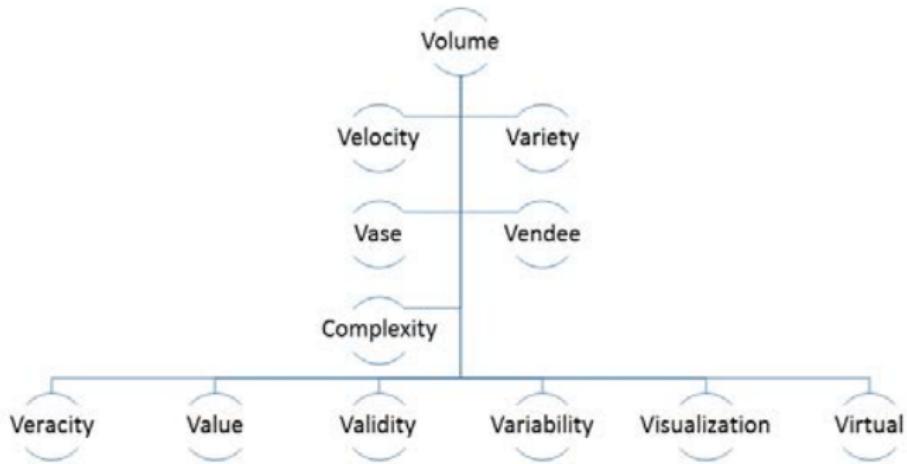


Figure: All V's of Big Data Paradigm

Total V's of Big Data

Technology



- ▶ Does the Big Data technology comply with all V's?
- ▶ Which V's are our research work?
- ▶ Is a Big Data technology perfect without considering all V's?
- ▶ Which V is more helpful in generating revenue?

Conclusion



- ▶ Introduced two new V's, namely, Vase and Vendee.
- ▶ Volume is defined with three new V, namely, Voluminosity, Vacuum, and Vitality.
- ▶ Finally, we have discussed $V_3^{11} + C$.

References I



- [1] Doug Laney, "3D Data Management: Controlling Data Volume, Velocity, and Variety", Gartner, file No. 949. 6 February 2001, <http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf>
- [2] M Ali-ud-din Khan, M F Uddin, and N Gupta, "Seven V's of Big Data: Understanding Big Data to extract Value", In 2014 Zone 1 Conference of the American Society for Engineering Education (ASEE Zone 1), pages 3-5, April, 2014, DOI: <http://dx.doi.org/10.1109/ASEEZone1.2014.6820689>
- [3] Eileen McNulty, "Understanding Big Data: The Seven V's", [online] Retrieved 10, June, 2016 from <http://dataconomy.com/seven-vs-big-data/>
- [4] Monica Bulger, Greg Taylor, and Ralph Schroeder, "Engaging Complexity: Challenges and Opportunities of Big Data", In London: NEMDOE, 2014.
- [5] C.L. Philip Chen, and Chun-Yang Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on Big Data", Information Science, 275(2014), pages 314–347, DOI: <http://dx.doi.org/10.1016/j.ins.2014.01.015>
- [6] Rob Livingstone, "The 7 Vs of Big Data – and by they are important for you and your business", [Online], published on 21, June, 2013, Retrieved on 10, June, 2016 from <https://livingstoneadvisory.com/2013/06/vs-big-data/>
- [7] landmark.solutions, "The 7 pillars of Big Data", A White Paper of Landmark Solutions, Retrieved on 10, June, 2016 from https://www.landmark.solutions/Portals/0/LMSDocs/Whitepapers/The_7_pillars_of_Big_Data_Whitepaper.pdf
- [8] "Internet Live Stats", [online], Retrieved on 11 June 2016 from <http://www.internetlivestats.com/total-number-of-websites/>
- [9] IDC, "Global smartphone shipments forecast from 2010 to 2020 (in million units)", [online], Retrieved on 12 June 2016 from <http://www.statista.com/statistics/263441/global-smartphone-shipments-forecast/>
- [10] Louis Columbus, "Roundup Of Analytics, Big Data & Business Intelligence Forecasts And Market Estimates, 2015", [online], Retrieved on 11 June 2016, <http://www.forbes.com/sites/louiscolumbus/2015/05/25/roundup-of-analytics-big-data-business-intelligence-forecasts-and-market-estimates-2015/#16f1c4834869>
- [11] Charles McLellan, "The internet of things and big data: Unlocking the power", [online], Retrieved on 10 jun 2016 from <http://www.zdnet.com/article/the-internet-of-things-and-big-data-unlocking-the-power/>

Thank You

Big Data

DR. RIPON PATGIRI

NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR

Definition

- Big Data is growing rapidly day-by-day due to producing a huge data from various sources.
- The new emerging technologies act as catalyst in growing data where the growth of data get an exponential pace.
- There are many V's coming up to define the characteristics of Big Data.
- Doug Laney defines Big Data using 3V's, namely, volume, velocity and variety.
- Now, the $V_3^{11} + C$ is used to define the characteristics of Big Data

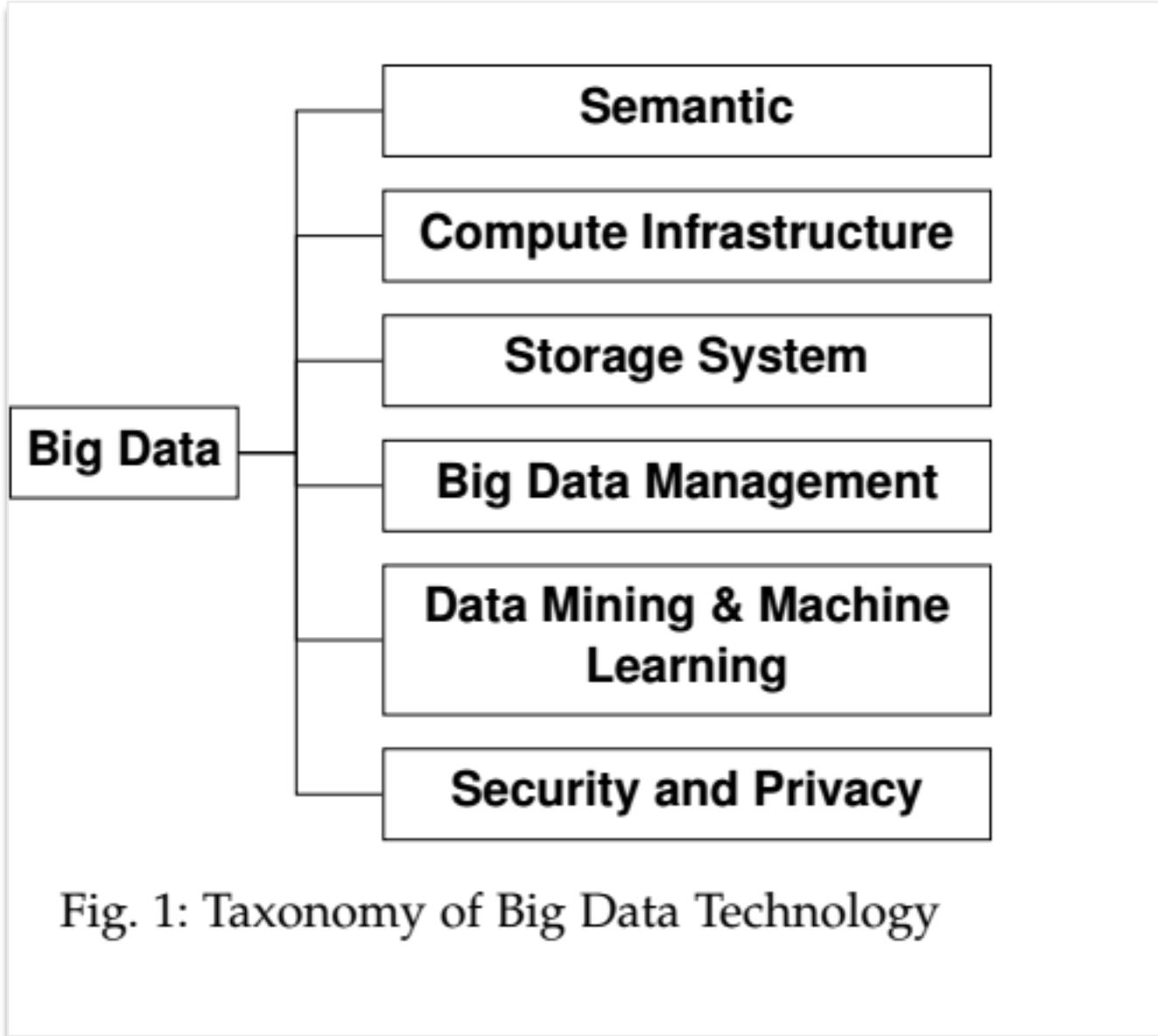


Fig. 1: Taxonomy of Big Data Technology

Taxonomy

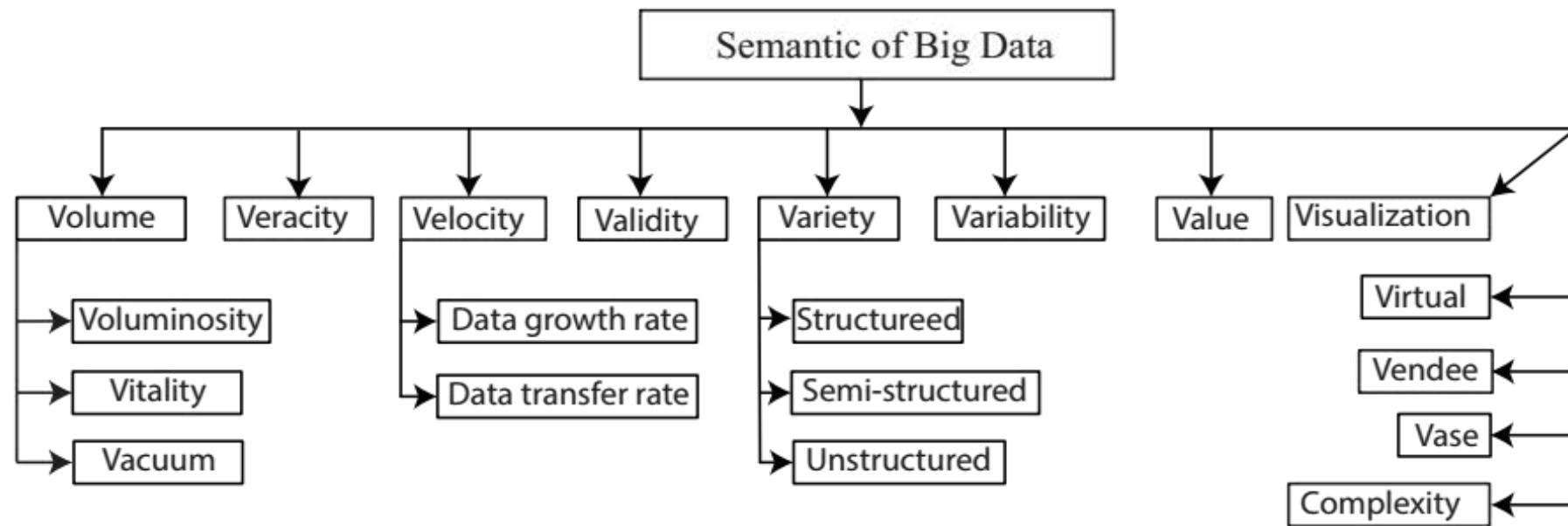


Fig. 2: Semantic of Big Data. Source [3]

Semantic

TABLE 1: Individual meaning the V family

Name	Short meaning	Big Data context
Volume	Size of data	Voluminosity, Vacuum and Vitality
Velocity	Speed	Transfer rate & Growth rate of data
Variety	Numerous types of data	Structured, unstructured and semi-structured
Veracity	Accuracy and truthfulness	Accuracy of Data
Validity	Cogency	Correct Data
Value	Worth	Giving worth to the raw data
Virtual	Nearly actual	Managing large number of data.
Visualization	To be shown	Logically display the large-set of data.
Variability	Change	Change due to time and intention
Vendee	Client management	Client management and fulfilling the client requirements
Vase	Big Data Foundation	IoT, Cloud Computing etc.
Complexity	Time and Space requirement	Computational performance

Description

Compute Infrastructures

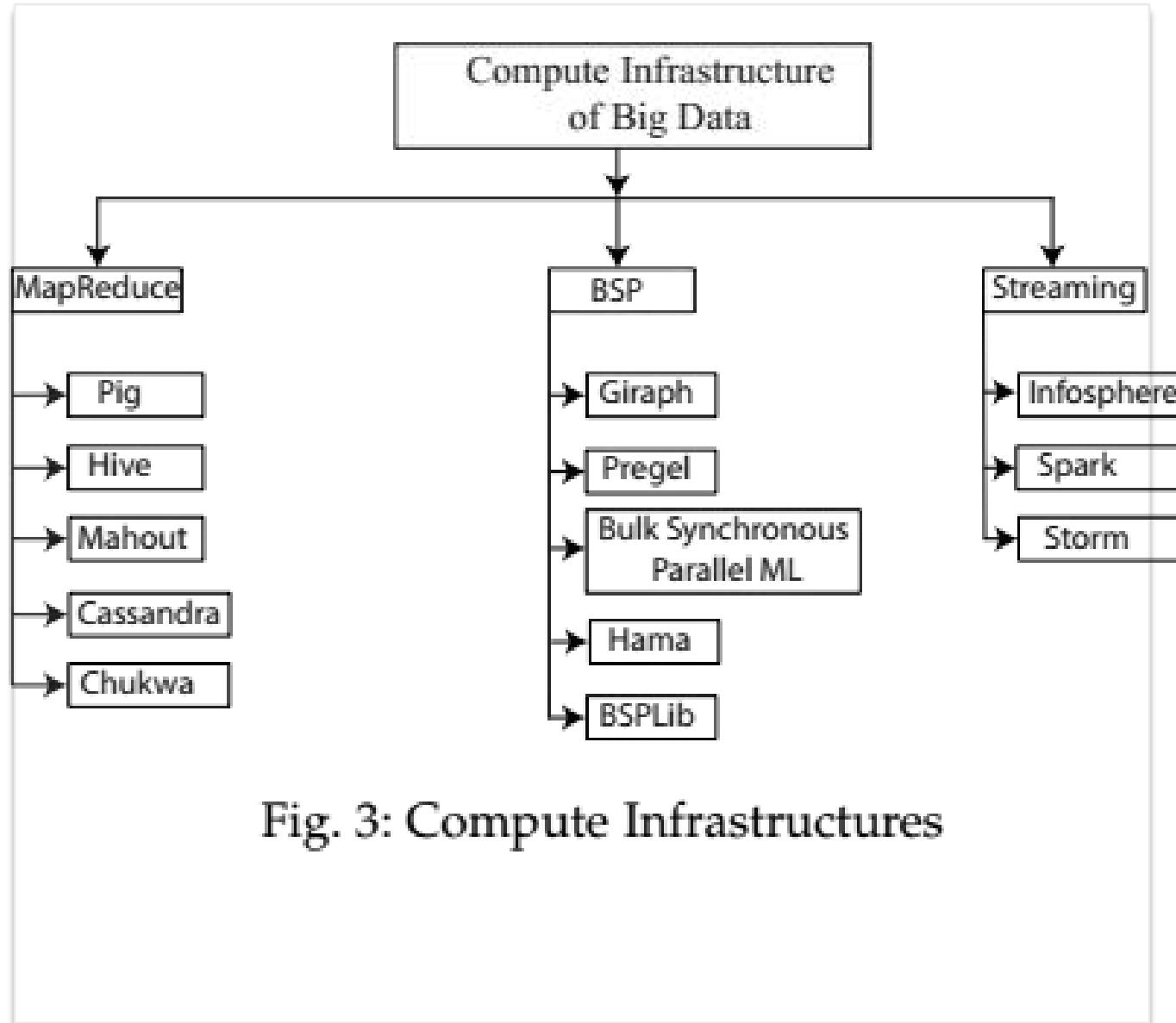


Fig. 3: Compute Infrastructures

MapReduce

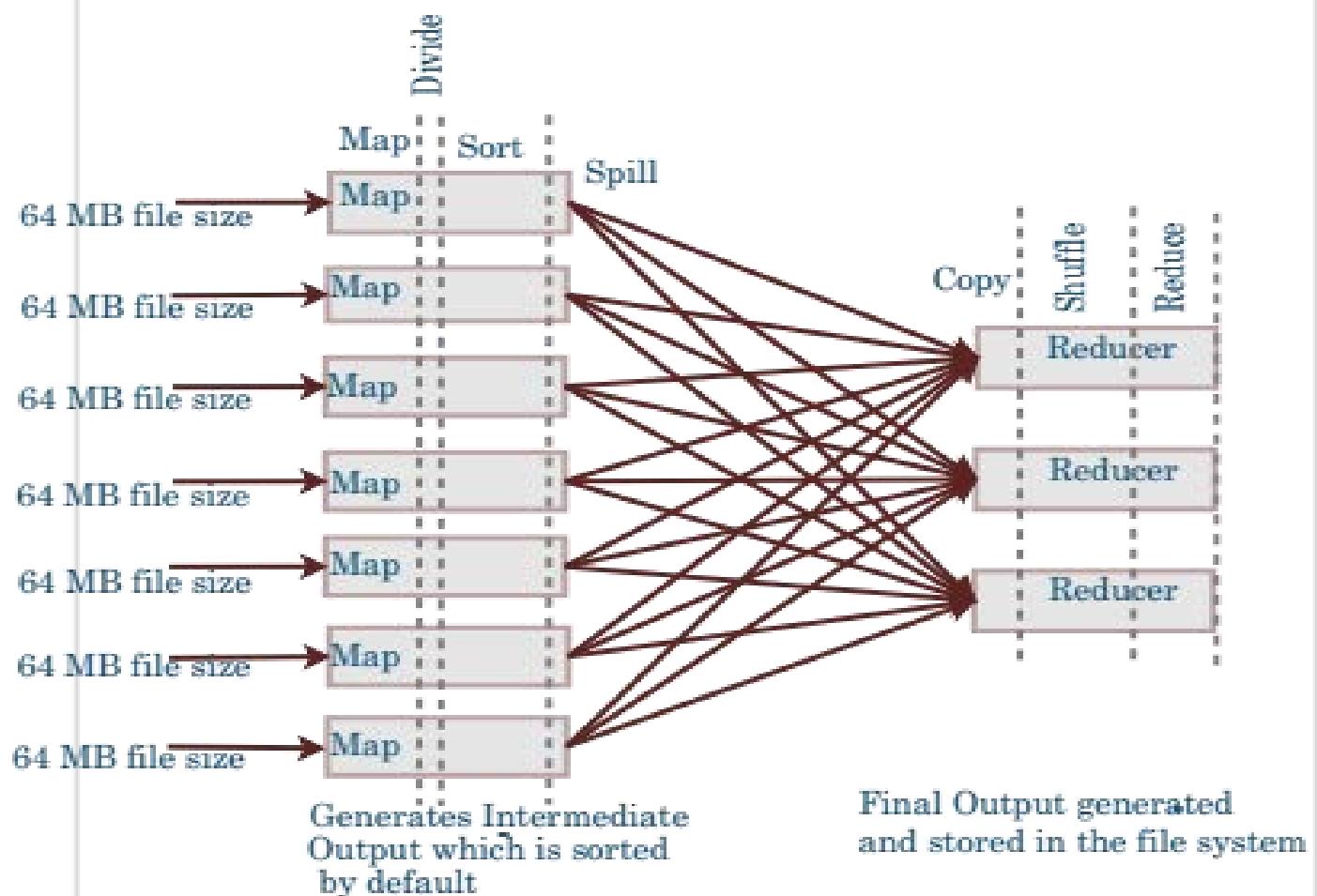
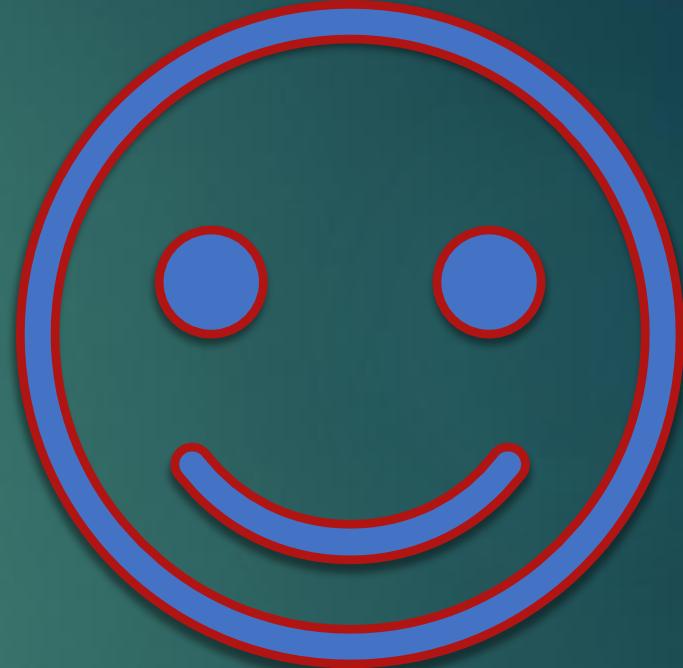


Fig. 4: Architecture of MapReduce

Thank You



Interconnected Cloud Computing Environments: Challenges, Taxonomy, and Survey

ADEL NADJARAN TOOSI, RODRIGO N. CALHEIROS, and RAJKUMAR BUYYA,
The University of Melbourne, Australia

A brief review of the Internet history reveals the fact that the Internet evolved after the formation of primarily independent networks. Similarly, interconnected clouds, also called *Inter-cloud*, can be viewed as a natural evolution of cloud computing. Recent studies show the benefits in utilizing multiple clouds and present attempts for the realization of an Inter-cloud or federated cloud environment. However, cloud vendors have not taken into account cloud interoperability issues, and each cloud comes with its own solution and interfaces for services. This survey initially discusses all the relevant aspects motivating cloud interoperability. Furthermore, it categorizes and identifies possible cloud interoperability scenarios and architectures. The spectrum of challenges and obstacles that the Inter-cloud realization is faced with are covered, a taxonomy of them is provided, and fitting enablers that tackle each challenge are identified. All these aspects require a comprehensive review of the state of the art, including ongoing projects and studies in the area. We conclude by discussing future directions and trends toward the holistic approach in this regard.

Categories and Subject Descriptors: D.4.7 [**Operating Systems**]: Organization and Design—*Distributed systems*; D.2.12 [**Software Engineering**]: Interoperability; C.2.4 [**Computer-Communication Networks**]: Distributed Systems

General Terms: Design, Standardization

Additional Key Words and Phrases: Cloud computing, cloud federation, Inter-cloud, multi-cloud, cross-clouds, utility computing

ACM Reference Format:

Adel Nadjaran Toosi, Rodrigo N. Calheiros, and Rajkumar Buyya. 2014. Interconnected cloud computing environments: Challenges, taxonomy, and survey. ACM Comput. Surv. 47, 1, Article 7 (April 2014), 47 pages.
DOI: <http://dx.doi.org/10.1145/2593512>

1. INTRODUCTION

Cloud computing is a term used to describe a paradigm for delivery of computing services to users on a pay-as-you-go basis. In this paradigm, users utilize the Internet and remote data centers to run applications and store data. The cloud technology allows more efficient computing by removing most of the upfront costs of setting up an IT infrastructure. It allows organizations to expand or reduce their computing facilities

Authors' addresses: A. N. Toosi, R. N. Calheiros, and R. Buyya, **Cloud Computing and Distributed Systems (CLOUDS)** Laboratory, Department of Computing and Information Systems, The University of Melbourne, Parkville Campus, Melbourne, VIC 3010, Australia. R. Buyya is also associated as a visiting professor for the University of Hyderabad, India; King Abdulaziz University, Saudi Arabia; and Tsinghua University, China. Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or permissions@acm.org.

© 2014 ACM 0360-0300/2014/04-ART7 \$15.00
DOI: <http://dx.doi.org/10.1145/2593512>

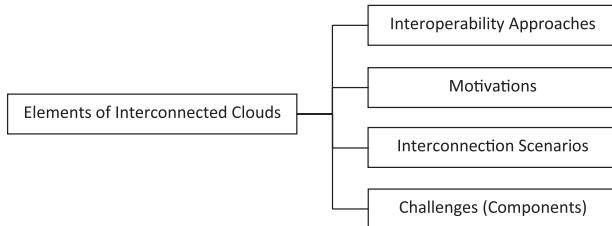


Fig. 1. Elements of interconnected cloud environments.

very quickly. There is an increasingly perceived vision that the birth of cloud computing is a big step toward the long-held dream of computing as a utility [Buyya et al. 2010].

Over the years, several technologies such as virtualization, grid computing, and service-oriented architecture (SOA) have matured and significantly contributed to make cloud computing viable. However, cloud computing is still in its early stage and suffers from lack of standardization. What actually happens is that most new cloud providers propose their own solutions and proprietary interfaces for access to resources and services. This heterogeneity is a crucial problem as it raises barriers to the path of the ubiquitous cloud realization. The main barrier is *vendor lock-in*, which is unavoidable at this stage [Rochwerger et al. 2009; Petcu 2011]; customers applying cloud solutions need to tailor their applications to fit the models and interfaces of the cloud provider, which makes future relocation costly and difficult. Furthermore, cloud computing, as a novel utility, requires ubiquitously interconnected infrastructure like other utilities such as electricity and telephony. Accordingly, interoperability and portability across clouds are important not only for protection of the user investments but also for realization of computing as a utility.

In this article, we aim to cover various aspects of interconnected cloud computing environments. Key elements of interconnected clouds are provided, and each one of them is classified in depth. Figure 1 shows the key elements of interconnected clouds from our point of view.

First, we consider interoperability approaches. Cloud interoperability in practice can be obtained through either *brokering* or *standard interfaces*. By using a service broker, which translates messages between different cloud interfaces, customers are able to switch between different clouds and cloud providers can interoperate. Standardization of interfaces is another common method for realization of interoperability. Part of this study covers different standards and initiatives related to technologies to regulate the Inter-cloud environment. However, one comprehensive set of standards is difficult to develop and hard to be adopted by all providers. A combination of the aforementioned approaches often occurs in practice.

If cloud interoperability happens, both cloud providers and customers benefit from different possible cloud scenarios as shown in Figure 2. Benefits of an interconnected cloud environment for both cloud providers and their clients are numerous, and there are essential motivations for cloud interoperability such as avoiding vendor lock-in, scalability, availability, low-access latency, and energy efficiency.

Cloud interoperability requires cloud providers to adopt and implement standard interfaces, protocols, formats, and architectural components that facilitate collaboration. Without these *provider-centric* changes, cloud interoperability is hard to achieve. Among different provider-centric approaches, *Hybrid Cloud*, *Cloud Federation*, and *Inter-cloud* are the most prominent scenarios. A hybrid cloud allows a private cloud to form a partnership with a public cloud, enabling the *cloud bursting* application deployment model. Cloud bursting allows an application to run in a private data center and to

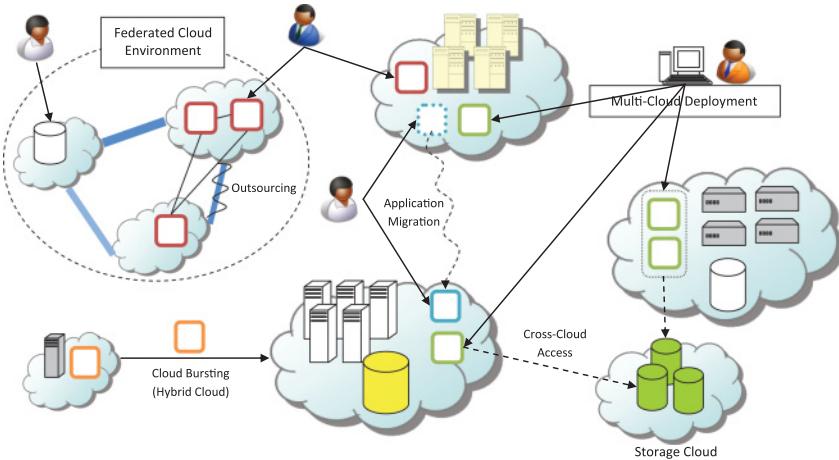


Fig. 2. Cloud interoperability and Inter-cloud.

burst into a public cloud when the demand for computing capacity spikes. Cloud federation allows providers to share their resources through federation regulations. In this paradigm, providers aim to overcome resource limitation in their local infrastructure, which may result in rejection of customer requests, by outsourcing requests to other members of the federation. Moreover, cloud federation allows providers operating at low utilization to lease part of their resources to other federation members in order to avoid wasting their nonstorable compute resources. Last but not least is Inter-cloud, in which all clouds are globally interconnected, forming a worldwide cloud federation. Inter-cloud removes difficulties related to migration and supports dynamic scaling of applications across multiple clouds.

Even if cloud interoperability is not supported by cloud providers, cloud customers are still able to benefit from *client-centric* interoperability facilitated by user-side libraries or third-party brokers. *Multicloud* application deployment using adapter layer provides the flexibility to run applications on several clouds and reduces the difficulty in migrating applications across clouds. *Aggregated service by broker*, a third-party solution in this regard, offers an integrated service to users by coordinating access and utilization of multiple cloud resources.

Cloud interoperability is a challenging issue and requires substantial efforts to overcome the existing obstacles. These include both functional and nonfunctional aspects. This study covers the spectrum of challenges in cloud interoperability. These challenges broadly cover security, service-level agreement (SLA), monitoring, virtualization, economy, networking, provisioning, and autonomies.

There are various projects and studies to propose and prototype approaches to enable interconnected cloud environments. They vary in terms of architecture, facilities they provide, and challenges they address. Another aim of this study is to survey and classify these projects.

In summary, the contributions of this article are:

- It introduces and analyzes the relevant aspects motivating cloud interoperability.
- It explores the spectrum of challenges and obstacles for Inter-cloud realization and identifies open challenges.
- It proposes a taxonomy of such challenges and obstacles and identifies enablers that tackle each of them.

- It presents a comprehensive review of the state of the art, including ongoing projects and research in the area.
- It discusses future directions and trends in the area.

This survey is organized as follows: In Section 2, we define and discuss several different terms and descriptions in the area of interconnected cloud environments to clarify the positioning of this work. In Section 3, we explore the motivation and benefits of interoperability between clouds. Then, we introduce different possible architectures for multiple cloud scenarios in Section 4. Classification of the challenges and potential enablers with respect to each challenge is provided in Section 5. Successful realization of cloud interoperation requires standards, so in Section 6, we review the current standard protocols and interfaces that facilitate the realization of cloud interoperation, including organizations and initiatives dealing with these standard technologies. In Section 7, we survey the state-of-the-art projects and developments in the area and fit them into taxonomies based on the characteristics and the challenges they address. We discuss different tools and frameworks supporting interoperable cloud infrastructure in Section 8. Finally, we conclude the study and provide a basis for future developments in this area.

2. TERMS, QUOTES, AND DEFINITIONS

In view of the fact that integration and aggregation of cloud services to achieve a seamless computing infrastructure have recently received attention and it is in the early stage of development, several different terms and descriptions have been used in the scientific community to define it. Precise understanding of these terms and definitions, including differences, similarities, and experts' comments, clarifies the current study and helps in future directions.

Inter-cloud has been introduced by Cisco [Bernstein et al. 2009] as an interconnected global “cloud of clouds” that mimics the known term Internet, “network of networks.” The Inter-cloud refers to a mesh of clouds that are unified based on open standard protocols to provide a cloud interoperability [Bernstein et al. 2009]. The main objective of the Inter-cloud is similar to the Internet model and telephone system, where everything is ubiquitously connected together in a multiple-provider infrastructure.

According Vint Cerf, vice president and chief Internet evangelist at Google, who is recognized as one of “the fathers of the Internet”:¹

... It's time to start working on Inter-cloud standards and protocols so your data doesn't get trapped in one of the problems with cloud computing ... [and these standards and protocols] allow people to manage assets in multiple clouds, and for clouds to interact with each other.

According to Gregory Ness, chief marketing officer at Vantage Data Centers, we can:²

think of the Inter-cloud as an elastic mesh of on demand processing power deployed across multiple data centers. The payoff is massive scale, efficiency and flexibility.

The Global Inter-cloud Technology Forum (GICTF), a Japanese organization that aims at promoting standardization of network protocols and interfaces through which cloud systems interwork with each other, defines Inter-cloud computing as:³

a cloud model that, for the purpose of guaranteeing service quality, such as the performance and availability of each service, allows on-demand reassignment of resources and transfer of workload through a

¹<http://www.guardian.co.uk/technology/blog/2010/feb/05/google-Cloud-computing-interCloud-cerf>.

²<http://Cloudcomputing.sys-con.com/node/1009227>.

³Use Cases and Functional Requirements for Inter-Cloud Computing: A white paper by Global Inter-Cloud Technology Forum (GICTF), http://www.gictf.jp/doc/GICTF_Whitepaper_20100809.pdf.

[sic] interworking of cloud systems of different cloud providers based on coordination of each consumer's requirements for service quality with each provider's SLA and use of standard interfaces.

The term cloud federation, on the other hand, implies the creation of a group of aggregated providers that are mutually collaborating to share their resources in order to improve each other's services [Kurze et al. 2011]. One definition of cloud federation can be deduced from Rochwerger et al. [2011], where the authors talk about the basic principles of cloud computing:

... federations of providers such that they can collaborate and share their resources. ... Any federation of cloud computing providers should allow virtual applications to be deployed across federated sites. Furthermore, virtual applications need to be completely location free and allowed to migrate in part or as a whole between sites. At the same time, the security privacy and independence of the federation members must be maintained to allow computing providers to federate.

Reuven Cohen, founder and CTO of Enomaly Inc.,⁴ defines cloud federation as follows:

Cloud federation manages consistency and access controls when two or more independent geographically distinct clouds share either authentication, files, computing resources, command and control or access to storage resources.

Cloud federation and Inter-cloud are relatively new in the cloud computing area. According to the aforementioned arguments, different standards are first required before it can be defined and subsequently realized.

The terms Inter-cloud and cloud federation are often used interchangeably in the literature. Similarly, we consider these as synonyms in this survey and contemplate Inter-cloud as a worldwide federation of the clouds. However, some practitioners and experts prefer to give different definitions to these terms. According to Ellen Rubin, founder and VP of Products at CloudSwitch,⁵ there are some key differences between Inter-cloud and cloud federation. The primary difference between the Inter-cloud and federation is that the Inter-cloud is based on the future standards and open interfaces, while federation uses a provider version of the interfaces. According to the discussion, cloud federation can be considered as a prerequisite toward the ultimate goal of the Inter-cloud. With the Inter-cloud vision, clouds must federate and interoperate and all would have a common perception of how applications should be deployed. In this model, interoperability of different cloud platforms is achieved without explicit referencing by user.

According to Chen and Doumeingts [2003], there are two distinguished approaches to obtain interoperability in practice:

- (1) Adhering to published interface standards
- (2) Developing a broker of services that can convert one product's interface into another product's interface "on the fly"

A relevant example of the first approach is TCP/IP and of the second kind of approach is enterprise application integration approaches, the CORBA architecture, and its object request broker (ORB) [Chen and Doumeingts 2003]. Current cloud federation and Inter-cloud projects, as expressed in Figure 3, follow either of the two methods or a combination of them. However, it seems that the Inter-cloud concept is typically close to the former approach, while the cloud federation idea is mostly inspired by the latter.

⁴Enomaly, <http://www.enomaly.com/>.

⁵<http://www.cloudswitch.com/page/Cloud-federation-and-the-interCloud>.

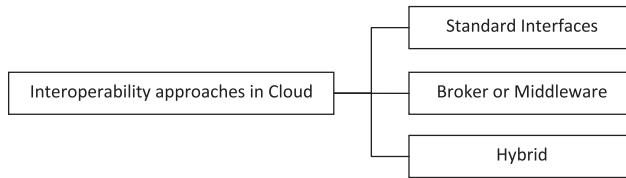


Fig. 3. Cloud interoperability approaches.

There is another group of professionals who believe that the aforementioned form of interoperability is unlikely to happen. They think of cloud integration as a completely separated layer that handles all the issues regarding aggregation and integration of the clouds as entirely detached from vendors and providers. We call this form of integration *multiple cloud, multicloud, or cross-cloud*. Aligning to this thought, Joe Skorupa, Gartner's⁶ vice president, said:⁷

Even if an open cloud standard should come to pass, every provider would still continue to implement its own proprietary enhancements to differentiate its wares from the competition. ... Vendors do not want clouds to become commodity products because they do not want to compete on price alone.

Interested readers can find a thorough survey and taxonomy of architectures and application brokering mechanisms across multiple clouds in an article by Grozev and Buyya [2012]. Their work is mostly focused on application-specific brokering mechanisms and can be positioned as a part of a broader view of interconnected cloud environments presented in this article.

In this section, we provided different terms and definitions including experts' quotes on interconnected clouds. Although there are different terms and titles, interoperability between clouds is the main requirement for the realization of these scenarios. Therefore, in this study, our aim is to cover all aspects in this regard. However, some discussions are just applicable for special cases, for example, cloud federation or multiclouds, which are clearly stated. In summary, we believe that transition toward Inter-cloud has already started and it is an inevitable need for the future of cloud computing. In this regard, we present supporting arguments and motivations for cloud federation and interconnected clouds in the next section.

3. MOTIVATIONS FOR CLOUD INTEROPERABILITY

Cloud computing has already provided considerable capabilities for scalable, highly reliable, and easy-to-deploy environment for its clients. Nevertheless, the benefits of interconnected cloud environments for both cloud providers and their clients are numerous and there are essential motivations for cloud interoperability, which will eventually lead to the Inter-cloud. In this section, key benefits of an Inter-cloud environment have been summarized as demonstrated in Figure 4.

3.1. Scalability and Wider Resource Availability

Even though one of the key features of cloud computing is the illusion of infinite resources, capacity in cloud providers' data centers is limited and eventually can be fully utilized [Calheiros et al. 2012a; Aoyama and Sakai 2011]. Growth in the scale of existing applications or surge in demand for a service may result in immediate need for additional capacity in the data center. Current service providers handle this issue by overprovisioning data center capacity. That is, the average demand of the system is

⁶Gartner, <http://www.gartner.com/>.

⁷<http://www.computerworld.com/s/article/9217158>.

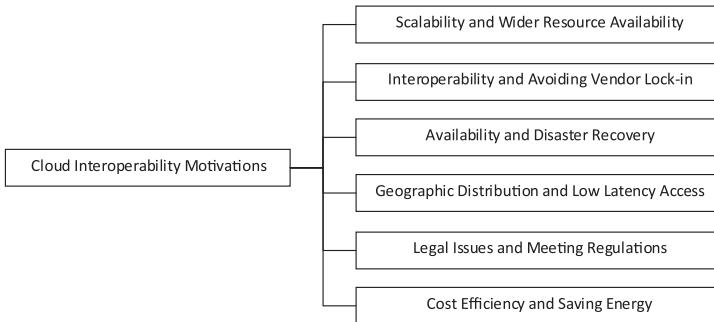


Fig. 4. Cloud interoperability motivations.

several times smaller than the capacity of its computing infrastructure. This strategy and the cost of its operation constitute a large expense for cloud owners. Actual usage patterns of many real-world application services vary with time and most of the time in unpredictable ways. Therefore, as we stated earlier, unexpected loads can potentially overburden a single cloud provider and lead to unreliable and interrupted services. It is overly restrictive in terms of small-size or private clouds. If cloud providers were able to dynamically scale up or down their data center capacity, they could save a substantial amount of money and overcome this issue. Scalable provisioning of application services under variable workload, resource, and network conditions is facilitated by interoperation of the clouds [Buyya et al. 2010]. Cloud federation helps the peak-load handling capacity of every enterprise cloud by resource sharing, without having the need to maintain or administer any additional computing nodes or servers [Rochwerger et al. 2009].

One may argue that public cloud providers are outstandingly elastic, with the perception of unlimited resources, so providers never need immediate additional capacity in their data center and they never fit into the aforementioned scenario. However, this claim does not obviate the need for additional capacity by small-size private clouds and for those applications requiring expansion across geographically distributed resources to meet quality of service (QoS) requirements of their users [Buyya et al. 2010].

3.2. Interoperability and Avoiding Vendor Lock-In

In economics, vendor lock-in is a situation where a customer becomes dependent on a vendor for its products or services and cannot move to another vendor without considerable cost and technical effort. It is also perceived as one of the current drawbacks of cloud computing [Armbrust et al. 2010]. With respect to cloud computing, vendor lock-in is the direct result of the current difference between the individual vendor paradigms based on noncompatible underlying technologies and the implicit lack of interoperability. Contemporary cloud technologies have not considered interoperability in design [Rochwerger et al. 2009; Bernstein et al. 2009]; hence, applications are usually restricted to a particular enterprise cloud or a cloud service provider. By means of cloud interoperability, cloud application deployment no longer needs to be customized. Cloud interoperability makes cloud services capable of working together and also develops the ability of multiple clouds to support cross-cloud applications [Bernstein et al. 2009].

3.3. Availability and Disaster Recovery

Although high availability is one of the fundamental design features for every cloud service, failure is inevitable. For instance, recently Amazon Web Services suffered

an outage, and as a result, a group of large customers dependent on Amazon were affected seriously.⁸ Unexpected failures can easily impose service interruption on a single cloud system. Aoyama and Sakai [2011] look into an instance of a service failure in which a cloud system witnesses a natural disaster. They identify the most important requirements for disaster recovery through cloud federation. In order to enable cloud providers to continue the delivery of guaranteed service levels even in such cases, a flexible mechanism is needed to relocate resources among the multiple cloud systems. Moreover, highly available cloud applications can be constructed by multiple cloud deployments to guarantee the required service quality, such as service availability and performance. Thus, cloud systems complement each other by mutually requesting required resources from their peers.

3.4. Geographic Distribution and Low-Latency Access

It is highly unlikely that a single cloud provider owns data centers in all geographic locations of the world to meet the low-latency access requirement of applications. Moreover, existing systems do not support mechanisms to dynamically coordinate load distribution among different cloud data centers. Since predicting geographic distribution of users consuming a cloud provider's services is not trivial, the load coordination must happen automatically, and distribution of services must change in response to changes in the load [Buyya et al. 2010]. Utilizing multiple clouds at the same time is the only solution for satisfying the requirements of the geographically dispersed service consumers who require fast response time. Construction of a federated cloud computing environment is necessary to facilitate provisioning of such application services. Consistently meeting the QoS targets of applications under variable load, resource, and network conditions is possible in such an environment.

3.5. Legal Issues and Meeting Regulations

Many cloud customers have specific restrictions about the legal boundaries in which their data or application can be hosted [Schubert et al. 2010]. Supplying resources in specific geographic locations to meet regulations in the places of those customers is an essential issue for a provider who wants to serve them. These regulations may be legal (e.g., an existing legislation specifying that public data must be in the geographic boundaries of a state or country) or defined by companies' internal policies [Calheiros et al. 2012a]. Cloud interoperability provides an opportunity for the provider to identify another provider able to meet the regulations due to the location of its data center.

3.6. Cost Efficiency and Saving Energy

The pay-as-you-go" feature of cloud computing directly awards economic benefits for customers by removing the cost of acquiring, provisioning, and operating their own infrastructures [Armbrust et al. 2010]. On the other hand, cloud computing providers should avoid the problem of the *idle capacity* (where their in-house hardware is not fully utilized all the time) and the problem of *peaks in demand* (where their own systems would be overloaded for a period). As the average demand of the system is several times smaller than the peak demand [Armbrust et al. 2010], providers are able to lease part of their resources to others, in order to avoid wasting their unused resources. Moreover, they can manage peaks in demand by purchasing resources from other underutilized providers. Both strategies help them to gain economies of scale, an efficient use of their assets, and enlargement of their capabilities through enhanced resources utilization [Celesti et al. 2010a]. Furthermore, this cooperation among cloud providers lowers the energy usage by promoting efficient utilization of the computing infrastructure.

⁸<https://cloudcomputing.sys-con.com/node/2416841>.

In a recent study done by Le et al. [2011], the plausibility of reducing cost and energy consumption by interconnecting cloud data centers has been investigated. They present a scenario in which a provider is able to save money by placing and migrating load across multiple geographically distributed data centers to take advantage of time-based differences in electricity prices. In addition, their policies reduce the required cooling power, considering data centers located in areas with widely different outside temperatures. In general, a unified interface that provides federated interoperation between clouds would help providers save costs and reduce their carbon footprint by energy-efficient utilization of physical resources.

4. CLOUD INTEROPERABILITY SCENARIOS

“Cloud computing refers to both the applications delivered as services over the internet and the hardware and systems software in the data centers that provide those services” [Armbrust et al. 2010]. In this definition, data center hardware and software are called *cloud* and the services can be sold in low-level abstraction like Amazon EC2⁹ or at a higher level like Google AppEngine.¹⁰ When a cloud is available to the public in a pay-as-you-go manner, it is called *public* cloud, and when a cloud belongs to a business or an organization and not made available to the public, it is called *private* cloud.

Cloud environments include a multitude of independent, heterogeneous, private, and public clouds. Based on Celesti et al. [2010a], the evolution of cloud computing can be hypothesized in three subsequent phases: *monolithic*, *vertical supply chain*, and *horizontal federation*. In the monolithic stage, cloud providers are based on their own proprietary architectures that create islands of cloud. Cloud services are delivered by different providers in this stage. In the vertical supply chain stage, some cloud providers leverage services from other providers. For example, an SaaS provider deploys services of an IaaS provider to serve its own customers. In horizontal federation, different-size cloud providers federate themselves to gain benefits of a cloud federation. For example, a fully utilized IaaS provider may use resources in an underutilized provider to accommodate more VM requests.

The main stakeholders in cloud computing scenarios are *cloud users* and *cloud providers* (CPs). Cloud users can be either software/application service providers (SPs) who have their service consumers or *end-users* who use the cloud computing services directly. SPs offer economically efficient services using hardware resources provisioned by CPs; that is, CPs offer utility computing service required by other parties. Different combinations of CPs and cloud users (SPs or end-users) give rise to a number of plausible scenarios between clouds [Ferrer et al. 2012].

According to this discussion, if interconnection happens between clouds at different levels of cloud stack layers (Figure 5), for example, a PaaS and IaaS provider, we call it *delegation* or *vertical federation*. But if interconnection between clouds happens at the same layer (e.g., IaaS to IaaS), we call it *horizontal federation*. Since the adoption of the latter faces many more hurdles, in this article we are mostly interested in the horizontal federation. Villegas et al. [2012] consider that a federated cloud structure can be viewed as a vertical stack analogous to the layered cloud service model. At each layer, a service request can be served either through local resources using delegation or by a partner cloud provider through federation.

If cloud interoperability requires cloud providers to adopt and implement standard interfaces, protocols, formats, and architectural components that facilitate collaboration, we call that *provider-centric* interoperability. Provider-centric scenarios are categorized as *hybrid* and *federated* cloud scenarios. In *client-centric* interoperability,

⁹Amazon EC2, <http://aws.amazon.com/ec2/>.

¹⁰Google AppEngine, <https://developers.google.com/appengine/>.

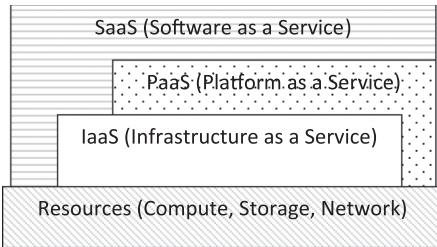


Fig. 5. Cloud service stack.

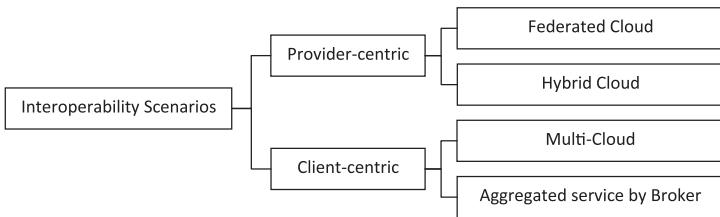


Fig. 6. Cloud interoperability scenarios.

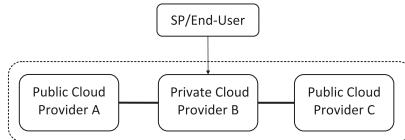


Fig. 7. Federated cloud scenario.

interoperability is not supported by cloud providers and cloud customers are required to initiate it by themselves or via third-party brokers. This kind of interoperability does not require prior business agreement among cloud providers and allows multiple cloud scenarios without adoption of common interfaces and protocols or with minimal adoption of the same. We consider *multicloud* and *aggregated service by broker* as client-centric interoperability scenarios. Figure 6 depicts the discussed classification, and the remaining parts of this section describe each scenario in detail.

4.1. Federated Scenario

In this scenario, SP establishes a contract with CP that itself is a member of a federation. A group of cloud providers are federated and trade their surplus resources among each other to gain economies of scale, efficient use of their assets, and expansion of their capabilities [Celesti et al. 2010a], for example, to overcome resource limitation during spike in demands. In this model, the computing utility service is delivered to SPs using resources of either one CP or a combination of different cloud providers. In such a scenario, the SP might be unaware of the federation and its contract is with a single cloud provider (Figure 7).

4.2. Hybrid Cloud Scenario

In hybrid cloud architecture, an organization that owns its private cloud moves part of its operations to external CPs. The organization can also sell idle capacity to other providers during periods of low load. This extension of a private cloud to combine local resources with resources from remote CPs is called hybrid cloud. In this scenario, an

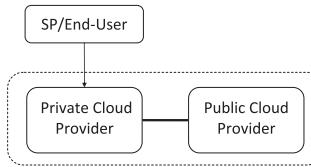


Fig. 8. Hybrid cloud scenario.

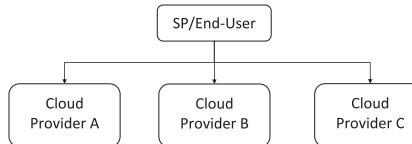


Fig. 9. Multicloud scenario.

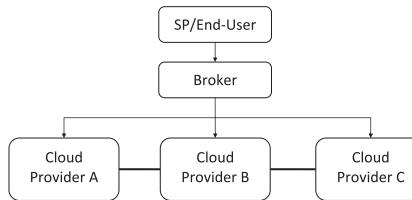


Fig. 10. Aggregated service broker scenario.

SP/end-user application can scale out through both private and public clouds when the local infrastructure is insufficient. Furthermore, this scenario can be extended if the organization offers capacity from its private cloud to others when that capacity is not needed for internal operations (Figure 8).

4.3. Multicloud Scenario

In this scenario, SPs or end-users are responsible to manage resources across multiple clouds. Service deployment, negotiating with each CP, and monitoring each CP during service operation are performed by the SP or end-user applications. In this case, the SP may require using an adapter layer with different APIs to run services on different clouds, or similarly, an end-user application may need a proper abstraction library. The important point about this scenario is that a separated layer handles all the issues regarding aggregation and integration of the clouds that is entirely apart from vendors and providers (Figure 9).

4.4. Aggregated Service by Broker

A new stakeholder, the broker, aggregates services from multiple CPs and offers an integrated service to the SPs or end-users. The deployment and management of components have been abstracted by the third-party broker. SPs or end-users benefit greatly from this model as the broker can provide a single entry point to multiple clouds. In this model, providers may also be required to install some internal components to support aggregated services by a trusted broker (Figure 10).

5. INTER-CLOUD CHALLENGES AND ENABLING APPROACHES

Inter-cloud raises many more challenges than cloud computing. In this section, the main challenges of cloud interoperability will be examined as listed in Figure 11.

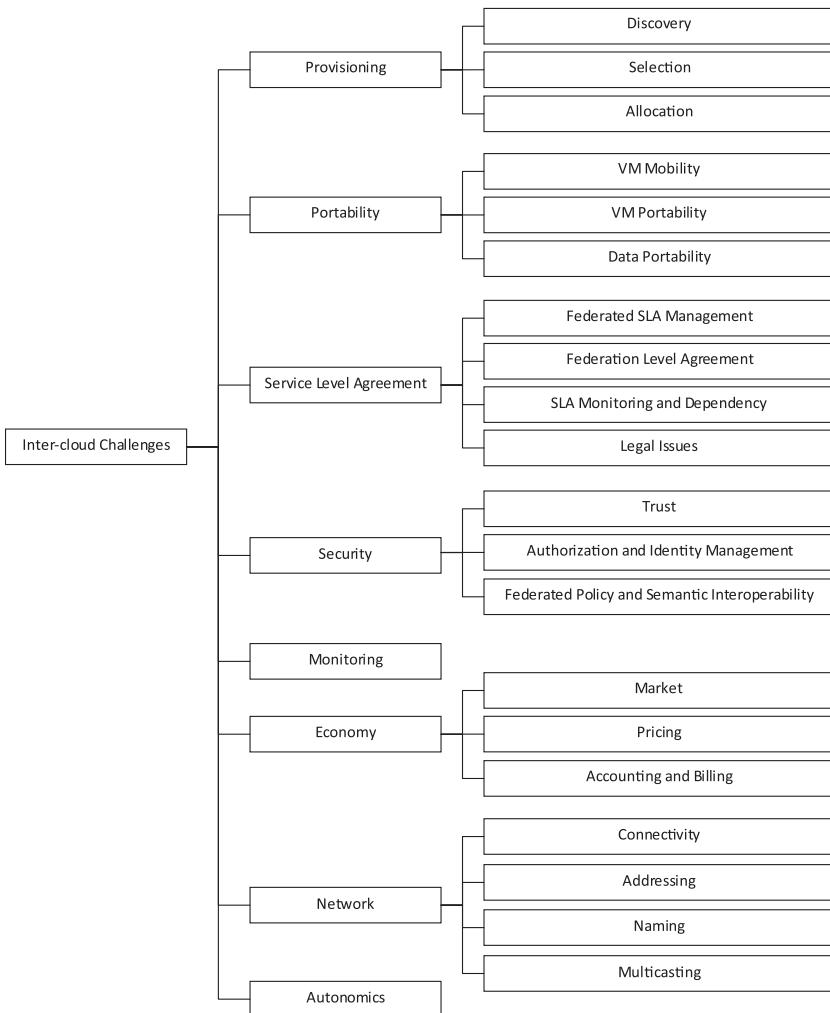


Fig. 11. Taxonomy of Inter-cloud challenges.

5.1. Provisioning

5.1.1. Discovery. Cloud service discovery allows automatic detection of services and resources offered by cloud providers on the Internet. Since cloud providers offer a variety of services and use different ways to describe them, a way to provide a common access to cloud services and to discover and deploy them is necessary. Cloud customers require selection of the best possible application deployments in the cloud according to their objectives and constraints for QoS. Achieving this goal requires effective discovery of the available services and their characteristics. In general, even though there are various forms of cloud services, discovery of services hosted in clouds has not received enough attention yet. For instance, Google App Engine and Amazon EC2 do not offer discovery services, and Microsoft Azure and Force.com offer limited discovery capabilities [Goscinski and Brock 2010].

One of the main issues regarding service discovery in a multiple-cloud deployment is the lack of an integrated repository of cloud services. Ranjan and Zhao [2011] believe

that centralized approaches for an integrated service catalog are not appropriate due to concerns of scalability, performance, and reliability arising from a large volume of service requests. They present a peer-to-peer cloud service discovery over a distributed hash table (DHT) overlay network. In contrast, Bernstein and Vij [2010a] argue that a point-to-point discovery architecture results in the n^2 complexity problem, and they propose *Intercloud Root Instances* and *Intercloud Exchanges* to solve the problem. In fact, Intercloud Root provides a service catalog, which is an abstracted view of the resources across disparate cloud environments.

Another issue is that cloud providers describe their services with diverse languages, terms, and names. Moreover, there is not a common understanding regarding service functionalities, their QoS, and metrics among providers and customers. In a heterogeneous environment such as Inter-cloud, it is difficult to enforce a standard syntax on service description or common metrics. Therefore, the use of syntactic-based approaches such as Universal Description, Discovery and Integration (UDDI)¹¹ is not applicable. Moreover, the Web Service Description Language (WSDL), which is used by UDDI, does not support modeling of QoS properties and it is difficult to add them. According to several recent studies [Bernstein and Vij 2010a; Moscato et al. 2010], the solution that covers mentioned drawback is a semantic web service that can increase expressiveness, flexibility, and accuracy of the service discovery by the application of ontologies for representation of the service properties. However, a detailed discussion about ontology-based approaches in this regard falls outside the scope of this survey.

Finally, states of a large part of services in clouds change constantly and are dynamic in nature. The situation is even worse in interconnected cloud environments. Consequently, dynamic attributes should be added to cloud services and a web service-based resource. A general framework for service and resource publication, discovery, and selection using dynamic attributes that expose current state and characteristics via web services has been proposed by Goscinski and Brock [2010].

5.1.2. Selection. Optimal application deployment in the cloud requires an effective selection strategy that works based on QoS criteria such as reliability, cost, and security and returns the set of the most suitable cloud services for end-customers. Cloud service selection did not receive much attention in the literature mostly due to the lack of reliable data on cloud services' QoS criteria.

Currently, selection is performed manually by cloud customers based on their requirements or through consultant companies. In multiple cloud application deployment scenarios, selection is not a trivial task due to the diversity of cloud services' characteristics and QoS. However, application deployment across multiple providers benefits from significant features such as the range of geographical locations, lower latency, higher reliability, lower deployment cost, higher failure resistance, and so forth. Consequently, an automated selection approach for application deployment is well motivated to optimize different aspects such as latency, reliability, throughput, data transfer, and cost. In addition, such a selection approach must take into account different constraints such as legal issues or security concerns.

The selection process can be performed either based on static information on the service quality provided by cloud providers or through dynamic negotiation of SLAs. Limited support is currently available for dynamic negotiation of SLAs [Petcu et al. 2011]. In order to design an infrastructure for negotiation and management of SLA in the cloud, several issues need to be addressed. A few cross-cloud projects (e.g., mOSAIC [Petcu et al. 2011]) focus on agent-based dynamic negotiation for cloud services.

¹¹A web-based distributed directory that enables providers to list their services and discover each other.

OPTIMIS, a toolkit proposed by Ferrer et al. [2012], focuses on cloud service and infrastructure optimization throughout three phases of the service life cycle: construction, deployment, and execution. In its deployment engine component, the OPTIMIS risk assessor provides multiple-criteria evaluation of cloud services using the Dempster-Shafer Analytical Hierarchy Process (DS-AHP).¹² The criteria that are used for evaluation can be generally classified as past performance, maintenance, security, customer support, and legal aspects. A value between 0 and 1 is computed for each of the criteria by evaluating a provider. These values are used as the basis for the final selection.

Another framework called CloudGenius has been introduced by Manzel and Ranjan [2012]. The framework provides a web application migration process and decision support. Cloud customers are able to migrate web applications to the cloud along a process that suggests cloud VM images and cloud infrastructure services according to requirements and goals of the cloud customer.

Han et al. [2009] present a cloud service selection framework in the cloud market that recommends best services from different cloud providers that match user requirements. Their framework ranks different services with providers and presents it to users so that they can select the appropriate or optimal services.

Resources and services in clouds can be represented by web services. There are considerable works in the context of SOA and grid web service selection. As a result, their contribution can be shared to tackle selection problem in clouds.

5.1.3. Allocation. Service selection on the customer's side leads to resource allocation on the provider's side. Resource allocation is a challenging issue from the cloud provider's perspective. Cloud providers usually offer their virtualized resources based on different QoS levels (e.g., best effort and reserved). Physical resources in clouds are shared between cloud users. Therefore, allocation strategies are needed to allocate resources to the requests in a profitable manner while fulfilling requests' QoS requirements.

As the number of resource consumers is increasing, clouds need to share their resources with each other to improve their quality of service. In general, such a collaborative cloud computing system (e.g., cloud federation) is prone to contention between user requests for accessing resources [Salehi et al. 2012]. Contention happens when a user request cannot be admitted or cannot acquire sufficient resources because resources are occupied by other requests (e.g., requests from the federated cloud provider). We call this issue "resource contention" from here onward.

Resource contention is not a new issue in federated environments. Various solutions have been proposed for the resource contention problem in federated cloud environments and other interconnected distributed computing systems [Salehi et al. 2012; Rochwerger et al. 2009; Toosi et al. 2011]. There is growing interest in the adoption of market-based approaches for allocation of shared resources in computational systems [Mihailescu and Teo 2010b]. Mihailescu and Teo [2010b] propose a dynamic pricing scheme for federated sharing of computing resources, where federation participants provide and use resources. They show that in their proposed dynamic scheme, the user welfare, the percentage of successful requests, and the percentage of allocated resources increase in comparison to the fixed pricing [Mihailescu and Teo 2010d]. Toosi et al. [2011] propose a model for trading of cloud services based on competitive economic models. They consider circumstances in which cloud providers offer on-demand and spot VMs while they participate in a federation.¹³ The resource manager unit

¹²The AHP is a structured technique for organizing and analyzing complex decisions and helps decision makers to find one that best suits their goal and their understanding of the problem. The DS theory is a mathematical theory that allows combining evidence from different sources to achieve a degree of belief.

¹³Spot VMs are VMs that can be terminated by providers whenever the current value for running such VMs (defined by the provider) exceeds the value that the client is willing to pay for using such resource.

evaluates the cost-benefit of outsourcing an on-demand request to a third party or allocating resources via termination of spot VMs. Their ultimate objective is to decrease the rejection rate and have access to seemingly unlimited resources for on-demand requests. Gomes et al. [2012] propose and investigate the application of market-oriented mechanisms based on the general equilibrium theory to coordinate the sharing of resources between clouds in the federated cloud. Goiri et al. [2011] propose an economic model that characterizes situations that assist decisions in a federated cloud, namely, when to outsource resources to other providers, when to admit requests from other providers, and how much capacity to contribute to the federation.

5.2. Portability

5.2.1. VM Mobility. The challenges regarding live virtual machine (VM) migration between physical nodes under the same administrative domain has been addressed previously in both industry and academia. The main challenge in this regard is that VMs require storage and network services from their hosts, and once a VM is live migrated from a host to another host, it still requires access to the storage and network services of the source host. Traditional support for live VM migration resolved this issue by a shared storage device and hosts, which are connected to the same local area network (LAN) [Nagin et al. 2011]. However, storage and network environments of different clouds are generally independent and are separated by firewalls.

VM Mobility is defined as the ability to move a running VM from one host to another without stopping it [Dowell et al. 2011]. In the Inter-cloud scenario, the cloud application might require VM Mobility. Moreover, Inter-cloud VM Mobility should not violate the independence of the respective clouds in terms of autonomy, privacy, and security. VM migration, from a source cloud to a destination one over a wide area network (WAN), constitutes transferring memory, status, and storage of the VM. According to Nagin et al. [2011], cross-cloud VM migration requires the following:

- (1) Memory and state transfer between hosts residing in different data centers
- (2) Same LAN access by VMs at the destination host, without two sites sharing LAN
- (3) Same storage access by VMs at the destination host, without two sites sharing storage

VM Mobility requires a VM transfer from one physical machine to another without disrupting the network traffic flow. Some hypervisors allow a running VM to migrate from one physical machine when they are connected to the same local area network (e.g., XEN and KVM). However, long-distance VM Mobility between sites with separate LANs in a federated cloud environment, as we stated earlier, must ensure that migrated VM will have access to the same LAN at the destination host, without a sharing LAN between two sites.

The suggested approach is extending LANs between sites using WAN encapsulation technologies, like F5, a global traffic manager, by VMware¹⁴ or VMware and Cisco Migration Solution.¹⁵ However, such LAN extensions may violate providers' IT infrastructure autonomy, security, and privacy requirements. A virtual networking technology is needed that will allow VMs connected to the same virtual network to communicate with each other over a private and isolated virtual network within and across clouds, systems like Vine [Keahey et al. 2009] and VNET [Sundararaj et al. 2004]. Regarding this issue, Nagin et al. [2011] propose proxy servers at the source and destination clouds that communicate with each other while hiding the details of the source and

¹⁴<http://www.f5.com/pdf/solution-center/f5-for-virtualized-it-environments.pdf>.

¹⁵http://www.cisco.com/en/US/solutions/collateral/ns340/ns517/ns224/ns836/white_paper_c11-557822.pdf.

destination hosts. In Section 5.6, we discuss challenges regarding VM Mobility in more detail.

5.2.2. Data Portability. Users or applications that store data in the cloud, especially for SaaS and PaaS applications, often require access to the data so that it can be used by services of other cloud providers. Giving users control over their data is an important part of establishing trust and is paramount for creating interconnected cloud environments allowing users to easily move their data from one cloud to another [Fitzpatrick and Lueck 2010]. If a cloud provider stores data in their own proprietary format, then users cannot move their data to other vendors without considerable cost and technical effort. Therefore, industry standards and exporting tools, or at the very least formats that are publicly documented, are required to avoid data lock-in. Nowadays, data portability is hindered by the lack of proper technology and standards and nonportability of the applications and data, which is exploited by cloud service providers for their own benefits [Petcu et al. 2013].

According to Hill and Humphrey [2010], solutions for avoiding data lock-in can be classified into the following categories:

- (1) Using APIs that have multiple independent implementations, for example, Amazon EC2 APIs, which are used by several others such as Eucalyptus [Nurmi et al. 2009];
- (2) Choosing a particular API that can run on multiple Clouds, for example, MapReduce and Hadoop;
- (3) Manually decoupling the cloud-specific code of the application designed for each cloud provider from the application logic layer;
- (4) Creation of widespread standards and APIs; and
- (5) Utilization of vendor-independent cloud abstraction layers such as jclouds¹⁶ and libcloud.¹⁷

In fact, the main obstacle for data movement in interconnected cloud environments is the lack of standard metadata and data formats. To deal with this issue, platform-independent data representation and standardization of data import and export functionality between providers are needed [Petcu et al. 2013].

Google [Fitzpatrick and Lueck 2010] attempts to address this problem through its *Data Liberation Front*,¹⁸ whose goal is to make it easier to move data in and out of Google products. This is a step toward what is called *data liberation* by Google and provides freedom of data movement between clouds. The data liberation effort focuses specifically on tools and methods that allow users to export any data they create and import into another service or competing products.

CSAL [Hill and Humphrey 2010] is a *Cloud Storage Abstraction Layer* to enable portable cloud applications and supports three storage abstractions: *Blobs*, *Tables*, and *Queues*. CSAL implementation provides a single unified view of cloud storage across platforms and manages the metadata necessary for utilizing storage services across multiple clouds.

Bernstein and Vij [2010cl] proposed the Simple Storage Replication Protocol (SSRP) for a federated cloud environment that facilitates distributed unstructured storage (e.g., Blobs) connectivity between clouds in a point-to-point manner.

Petcu et al. [2013] proposed APIs for the mOSAIC project [Petcu et al. 2011] that provides portable cloud application development solutions.

¹⁶jclouds, <http://code.google.com/p/jclouds/>.

¹⁷libcloud, <http://libcloud.apache.org/>.

¹⁸Data Liberation Front, <http://www.dataliberation.org/>.

Bermbach et al. [2011] present MetaStorage, a federated cloud storage system that replicates data on top of diverse storage services using scalable distributed hash tables.

5.3. Service-Level Agreement

5.3.1. Federated SLA Management. Cloud providers define (or negotiate with customers) a service-level agreement (SLA) to specify what they guarantee. In a simple definition, SLA is a contract that describes a service and, most importantly, sets the expected service-level objectives (QoS expectations). It can even encompass more details such as penalties applied to the provider if it does not deliver services according to the service-level objectives. Implementation of SLA mechanisms on top of federated resources is still an open question. Since it is an area whose volume of ongoing works enables a survey of its own, we are not intended to cover all issues related to the SLA management in this article.

In federated cloud environments, it is expected that each participant cloud provider has its own SLA management mechanisms. Since user applications in such an environment exploit services and resources from different providers, one role of the federation is to set up and enforce a global SLA. By global SLA, we mean comprehensive SLAs between the user and the federation including all SLAs for each cloud provider. The federation should monitor the application in order to verify that the SLA is met by providers and should react to SLA violations.

In federated cloud environments, the entity that acts as a mediator between the cloud consumer and interoperable cloud providers must select services from different providers that better meet the user requirements. In such a dynamic environment, cloud providers can offer or meet guarantees according to their resource situation at the time the service is requested. Moreover, the service provided for the user might be composed of services from different providers. Hence, methods and protocols for negotiation of dynamic and flexible SLAs is a must for dynamic environments such as Inter-cloud. This demands that agreements are established dynamically at the time the service is requested, rather than advertised as an invariant property of the service.

A similar scenario happens when a broker of service acting on behalf of the user selects services among multiple cloud providers. There are several proposals addressing the negotiation of dynamic and flexible SLAs in service-oriented environments including *WS-Agreement* [Andrieux et al. 2004] and *WS-Agreement Negotiation* [Battré et al. 2010]. *WS-Agreement* provides language and protocols for creation agreements based on offers and for monitoring of agreement compliance at runtime. *WS-Agreement* supports a one-round negotiation process, whereas *WS-Agreement Negotiation* can be used when a more flexible and dynamic negotiation process is required. *WS-Agreement Negotiation* provides renegotiation capabilities on top of the *WS-Agreement* specification.

Another important issue in the federated cloud environment is how SLAs can be enforced in a federation where there are conflicting policies and goals of different members versus the objectives of the federation as a whole. For example, the federation layer can offer an SLA that promises highly reliable service, while none of the federation members, which are self-interested parties trying to maximize their revenue, are willing to offer such a service, which is costly.

There are a few works in the literature that addressed SLA management in the context of the federated cloud environment. Contrail [Carlini et al. 2012] is a project that proposes a federated and integrated cloud architecture. They provide extended SLA management functionalities by integrating the SLA management approach of the SLA@SOI¹⁹ project in the federation architecture. The Contrail federation coordinates the SLA support of different cloud providers. As cloud providers in the cloud federation

¹⁹SLA@SOI, <http://sla-at-soi.eu/>.

have their own SLA management mechanisms, Contrail tries to set up, coordinate, and enforce a federation-level SLA. In the context of the mOSAIC project [Petcu et al. 2011], there also exists facilities in order to offer user-oriented SLA services to final users [Amato et al. 2012]. Cuomo et al. [2013] propose an SLA-based broker operating in a volunteer computing environment to discover the most suitable resources to a user when resources are not reliable and can provide QoS comparable to those offered by commercial cloud providers.

5.3.2. Federation-Level Agreement. In addition to SLA, there can be a contract—so-called Federation-Level Agreement (FLA)—that includes the set of rules and conditions that has to be signed by new providers once they join the federation [Toosi et al. 2011]. For example, a federation can set rules for minimum resources contributed to the federation or the set of QoS such as minimum expected availability. In a more complex scenario, the federation can set different rules to have multiple pools of resources with different QoS guarantees.

5.3.3. SLA Monitoring and SLA Dependency. In federated cloud environments where a provider outsources its load to another provider, it expects a set of guaranteed QoS that is compatible to the promised QoS to end-users. Therefore, either the federation has to provide a service to match end-user QoS requirements or cloud providers have to do it on their own. In addition, in the cloud federation environment, there might be dependencies between performances of services provided by different providers. As explained in the “Practical Guide to Cloud Service Level Agreements by Cloud Standards Customer Council,”²⁰ there can be an environment where a cloud user has an SLA with a provider, and the cloud provider by itself has SLAs with two cloud providers and utilizes their resources to provide services to the end-user. Therefore, quality of a service can be affected by external services. It means that if one of the lower-layer services (e.g., infrastructure layer) is not functioning properly, it can affect the performance of higher-layer services (e.g., software layer). A practical approach is required to model the dependencies among services.

Winkler et al. [2010] propose an approach for automated management of SLAs to support composite services. In that approach, the explicit knowledge about a set of dependencies to automate the tasks of negotiation and renegotiation of SLAs and the handling of service-level objective (SLO) violations are taken into account.

Bodenstaff et al. [2008] propose an approach called MoDe4SLA to monitor dependencies between SLAs when managing composite services. In that case, different types of dependencies between services and the impact that services have on each other are analyzed during the development phase. The approach has not provided an Inter-Cloud language to share common understanding regarding the QoS criteria and their measurement units.

Dastjerdi et al. [2012] show how dependency knowledge can be modeled using semantic technology and how that knowledge can be used in discovery of monitoring services and SLA failure detection. The major contributions of the work are modeling services’ performance interdependencies and elimination of SLA failure cascading effects on violation detection.

5.3.4. Legal Issues. Interconnected cloud computing environments extricate applications from being confined to a single data center and open opportunities for globalizing and integrating services from multiple and disparate geographies. Besides technical complexities of interconnecting clouds, legal issues might arise with the realization of

²⁰Practical Guide to Cloud Service Level Agreements: A white paper by the Cloud Standards Customer Council, http://www.cloudstandardscustomerouncil.org/2012_Practical_Guide_to_Cloud_SLAs.pdf.

Inter-cloud. Cloud computing by itself requires consideration of a broad set of legal issues such as privacy and security, contracting issues, issues related to location and ownership of data, and business considerations [Bowen 2010]. Inter-cloud makes existing legal issues more complicated and also introduces new ones. A major part of these issues requires defining new laws and regulations beyond technical innovations and falls out of the scope of this survey. Readers who are interested in where legal issues might arise in cloud federation are referred to Kertesz and Varadi [2014].

When different organizations are involved in providing services for customers, one major issue is that it is difficult to guarantee confidentiality and privacy on data, especially when data is located in different countries with different laws. Cloud providers in interconnected cloud environments must provide mechanisms to guarantee the security and privacy of sensitive data within legal borders. For example, in a federated scenario in which a cloud provider leverages another provider's services and customers might not generally have control of where the service they are leasing is operating, in case of failures in the service delivery it will be difficult for the cloud user to identify the real causes. Moreover, legislation and laws concerning the privacy and security of data are different among different countries, and even among different states within the same country. For instance, based on the European Union (EU) directive, any personal data generated within the EU is subject to the European law and data cannot leave the EU unless it goes to a country that provides an adequate level of protection [Bowen 2010]. The US PATRIOT Act²¹ allows the U.S. government to gain access to personal financial information and student information stored in electronic systems just by providing a governmental certificate that the information might be relevant to criminal activities [Bowen 2010].

From a technical point of view, in interconnected cloud environments, application deployment requires that juridical and legislative restrictions be considered. Therefore, geo-location and legislation awareness policies must be imposed into the entity that acts as a mediator between the cloud consumer and interoperable Cloud providers (e.g., broker or federation layer) [Grozev and Buyya 2012], and compliance with such policies and agreement must be enforced. For instance, as part of the SLA management system, services of specific vendors can be avoided or placing the data outside a given country can be prohibited.

Management of legal issues for Inter-cloud requires defining a comprehensive body of laws compliant with all the legislation of the countries involved in possible transactions among cloud providers and users. This falls beyond technical aspects and constitutes efforts by legislatures to facilitate Inter-cloud by defining proper laws and regulations.

5.4. Security

5.4.1. *Trust*. In a social context, trust typically refers to a situation where one party is willing to rely on the actions of another party. The control over the actions is abandoned by the former to the latter. Ultimately, there is uncertainty as to whether the trusted party will behave or deliver as promised.

In the cloud computing environment, customers must trust in a cloud provider for the privacy and security of their assets (i.e., their data and processes). The degree of lost control over the data and processes depends on the cloud service model. In cloud computing, the risk of losing data confidentiality, integrity, and availability for customers is triggered by the lack of control over the data and processes [Khan and Malluhi 2010].

²¹Uniting and Strengthening America by Providing Appropriate Tools Required to Intercept and Obstruct Terrorism Act of 2001 (USA PATRIOT Act).

In interconnected cloud environments, establishment of trust is even more important and complex [Bernstein and Vij 2010b], as besides the customers, cloud providers must trust each other. In the Inter-cloud scenario, the trust and reputation of a cloud provider affect other cloud providers. Specifically, as Inter-cloud computing constitutes collaboration between independently owned autonomous clouds, and these providers peer with each other to outsource requests or data, there is a need for mechanisms to evaluate the trustworthiness of a federation member [Abawajy 2009].

Apart from reputation-based trust, because of the criticality of many computing tasks and diversity and vastness of services in Inter-cloud environments, formal trust mechanisms are required to help Inter-cloud entities (i.e., providers and users) to trust each other. A formal process for assessment of cloud services and their providers in such a highly dynamic system requires independent third parties that are acceptable to all entities. The concept of *trust federations* can be used in this regard. Trust federation is a combination of technology and policy infrastructure that allows organizations to trust each other's verified users to enable the sharing of information, resources, and services in a secure and distributed way.

The International Grid Trust Federation (IGTF)²² is a prominent example of a trust federation that can be leveraged for the Inter-cloud scheme as well. The IGTF is an organization that fosters harmonization and synchronization policies with the goal of enhancing establishment of cross-domain trust relationships for intergrid participants. Grid entities use X.509 certificates for authentication and authorization. These certificates are issued by the Certificate Authorities (CAs) that are part of the IGTF. In order to ensure compliance with established policies and guidelines, these CAs should be externally audited periodically. The IGTF has established such sets of policies and guidelines and ensures compliance to them between its members.

If we assume that the challenges regarding the trust evaluation of an Inter-cloud provider has been overcome and a provider is able to find a trustable party to peer with, we still face the challenge of building a trusted context for interconnected providers. In this trusted context, providers must be able to access each other's services while they still adhere to their internal security policies.

Currently, public key infrastructure (PKI) is the common trust model in cloud environments [Bernstein and Vij 2010b]. A PKI is a system that verifies a particular public key belongs to a certain entity. The process is done through the creation, storage, and distribution of digital certificates. The PKI creates digital certificates that map public keys to entities, stores these certificates in a central repository securely, and revokes them if needed.

The current PKI certificates-based trust model only checks if the entity is either trusted or nontrusted. However, an all-or-nothing trust model is not appropriate for the Inter-cloud environment in which cloud providers may have different levels of trust in each other [Bernstein and Vij 2010b]. Thus, they suggest a dynamic trust-level model layered on top of the PKI certificate-based trust model. Trusted context has been previously investigated in other collaborative environments. However, customized enabling technologies such as XACML²³ and SAML²⁴ were also proposed to build a trusted context for cross-cloud federation [Celesti et al. 2010a]. Abawajy [2009] proposes a distributed framework that enables parties to determine the trustworthiness of other entities. The proposed trust framework is a reputation-based trust management system that enables a service requester to obtain the trustworthiness of the service.

²²International Grid Trust Federation (IGTF), <http://www.igtf.net/>.

²³eXtensible Access Control Markup Language (XACML), OASIS, <https://www.oasis-open.org/committees/xacml/>.

²⁴Security Assertion Markup Language (SAML), OASIS, <https://www.oasis-open.org/committees/security>.

5.4.2. Authorization and Identity Management. Identity Management (IdM) is an administrative task that deals with authentication of individuals in a system and authorization to access resources of the system based on the associated rights and restrictions. In cloud computing environments, identity management services are mainly responsible for supporting access control to services based on user attributes (e.g., IP address, user and group name) and resource attributes (e.g., availability periods).

Identity management systems, in federated cloud environments, should allow identification of users and resources in addition to support interoperability across multiple identity domains [Núñez et al. 2011]. In such a scenario, users should be able to access various resources and services offered by different service providers once they are successfully authenticated in the Inter-cloud interface [Núñez et al. 2011]. One of the problems related to this issue is how to free users from the burden of authenticating with resources from multiple cloud providers. In other words, since each cloud has its own authentication mechanism, a standard method that provides Single Sign-On (SSO) authentication within Inter-cloud environments should be deployed. This must be applied both for customer-provider and provider-provider interactions.

In a federated environment, the SSO issue can be achieved through the *delegation of trust* that allows an entity to act on another entity's behalf. This is especially important when resources and services of different service providers are involved in serving an Inter-cloud application and it might be redundant or very expensive to authenticate each and every time a user or application has to access the resource. Utilizing proxy certificates is a common way of delegating trust that is successfully used in grid computing and service computing. This method allows entities of a federated system to securely interact and might require multiple levels of delegation by establishing a *chain of trust* of proxy certificates. SSO can also be achieved through the use of a trusted third party who will certify credentials on behalf of all parties in the federation. In fact, instead of knowing all possible entities, it is enough to be able to verify claims from the trusted third party.

Effective identity management in Inter-cloud environments requires support for established standards such as X.509 certificates, SAML, and WS-Federation [Bernstein and Vij 2010b]. These standards use different “languages” to express the identity information. A thorough solution is required to deal with these incompatibilities. Moreover, to solve the problem of different formats, names, and meanings for identity attributes, identification of common attributes or use of ontology is also suggested [Núñez et al. 2011]. Consequently, interoperability of identity management systems is a key issue that has to be taken into account.

Another issue that must be considered is how to manage the life cycle of identities. Typically, in the Inter-cloud environment, digital identity information is required in many directories and data stores, but it is hard to keep them synchronized with each other and remove or disable entries when required. In this direction, Service Provisioning Markup Language (SPML) proposed by OASIS is a possible solution [Núñez et al. 2011].

Celesti et al. summarize the requirements of Inter-cloud Identity Management in two categories [Celesti et al. 2010a]:

- (1) Single Sign-On (SSO) authentication, where a cloud must be able to authenticate itself to gain access to the resources provided by federated foreign clouds belonging to the same trust context without further identity checks
- (2) Digital identities and third parties, where a cloud has to be considered as a subject distinctively identified by credentials and each cloud must be able to authenticate itself with foreign clouds using its digital identity guaranteed by a third party

They propose an Inter-cloud Identity Management Infrastructure according to the selected technologies ranging from XMPP and XACML to SAML [Celesti et al. 2010c]. Support of XACML-compliant entitlement management is highly desirable for the Inter-cloud environment [Bernstein and Vij 2010b]. XACML provides a standardized language and method of access control and policy enforcement.

Another main problem that must be taken into account for Identity Management in large interconnected cloud environments is *scalability*. Performance of any IdM needs to be scalable and operation must be agile and quick. With the current technologies for IdMs, security must be compromised in favor of scalability. Methods such as PKI perform based on the *top-down* approach, where each entity starts out knowing the Root Certificate Authority (CA) and retrieves all the certificates from the Root down to its own key. Root CA is the only trusted body to certify name-to-key mappings. However, for scalability purposes, we require a hierarchy of CAs to be used. Approaches such as *Friend-of-a-Friend (Foaf)*²⁵ that do not rely on the root of trust can be helpful in this regard. Foaf provides machine-readable ontology for describing persons, their activities, and their relations to other people and objects and commonly used in social networks. Although methods like Foaf provide less strong security, they are highly scalable and obviate the need for a centralized database.

All in all, Federated Identity Management (FIM) is a critical step toward the realization of Inter-cloud. Identity federation can be accomplished in a number ways, from use of formal Internet standards, such as SAML and XACML specifications, to open-source technologies or other openly published specifications, such as OpenID,²⁶ OAuth,²⁷ and WebID.²⁸

Cloud computing provides on-demand resource provisioning using vast amounts of data and computing resources in centralized data centers; it was not designed based on the idea of federating distributed computing resources in geographically dispersed locations like a grid computing environment. Most of the security concerns in interconnected cloud environments underlie the coordinated resource sharing and problem solving in dynamic multiple administrative domains. The concept of *Virtual Organizations (VOs)*²⁹ defined in grid computing is highly relevant in this regard. For instance, Makkes et al. [2013] present the Inter-cloud Federation Framework that attempts to reuse successful experiences of VOs within grids.

5.4.3. Federated Policy and Semantic Interoperability. Various cloud providers may collaborate in Inter-cloud environments to provide aggregated services for clients. In other words, providing service for the application might consist of multiple services from different providers. These cloud providers can have different privacy mechanisms and security approaches. This heterogeneity must be addressed, and mechanisms are necessary to securely handle such a dynamic collaboration and authorization to use resources (data, services, etc.) during the interoperation process. Hence, providers should carefully manage access control policies and should agree upon some well-understood common *federated policy* to ensure that integration does not lead to any security breaches [Takabi et al. 2010].

Achieving such agreements requires some degree of *semantic understanding* or *semantic interoperability* due to the domain heterogeneity and different access policies each service provider offers. Solutions like Shibboleth [Needleman 2004], VOMS

²⁵Friend-of-a-Friend (Foaf) project, <http://www.foaf-project.org/>.

²⁶OpenID, <http://openid.net/>.

²⁷OAuth, <http://oauth.net/>.

²⁸WebID, <http://webid.info/>.

²⁹Virtual Organization (VO) is defined as a collection of individuals and institutions that access and share resources for the purpose of one or more identified goals within the grid.

[Alfieri et al. 2004], or XACML provide different methods to enable administrators to apply authorization policies; however, these approaches are designed to describe security policies and they are not able to handle the semantics associated with the elements they are protecting. Moreover, they do not properly cope with policy conflicts nor detect risky security situations as a result of errors in definition of complex policies. Semantic approaches for policy representation provide the ability to analyze policies related to entities described at different levels of abstraction.

In order to achieve a high level of expressiveness for policy definitions, Pérez et al. [2011] present an authorization architecture that relies on semantic web technologies. Heterogeneity of multiple organization domains and support for different policies have been taken into account by providing the capability to describe the semantics of the resources that are to be protected. Hu et al. [2009] propose a semantic access control approach applying semantic web technology to access control in cloud computing environments. Singhal et al. [2013] present a generic cloud collaboration framework that allows cloud user applications to use services from multiple clouds without prior business agreements among cloud providers and without adoption of common standards.

In summary, security challenges related to interconnected cloud environments are numerous, and it is not in the scope of this article to cover all the security-related issues of this area. There is a wealth of literature dealing with security aspects in distributed environments (e.g., grid), which are closely connected to interconnected cloud environments. Hence, many Inter-cloud security-related issues can be addressed based on the experiences in grid computing. Interested readers are referred to Bernstein and Vij [2010b], Chakrabarti et al. [2008], and Singhal et al. [2013] for more details regarding security concerns and challenges in interconnected cloud environments.

5.5. Monitoring

Cloud monitoring is a broad term that means monitoring of various aspects of the service, from VM performance to a very complicated monitoring of mutually dependent services in the cloud. Monitoring systems are required to monitor performance of physical and virtual resources and running cloud applications. A monitoring system can audit and aggregate data to help an administrator or a service manager to make sure the applications and contents are performing properly. In other words, the monitoring system gathers data from all the components within the cloud architecture and provides the data for the infrastructure and service management. Monitoring data is used for different purposes such as enforcing SLAs, enabling elasticity, ensuring QoS, and so forth.

Monitoring of cloud-based applications can be achieved in two separate levels of the *infrastructure* and the *application*. Infrastructure-level resource monitoring aims at measuring and reporting system parameters related to the infrastructure services offered to the user such as CPU, RAM, or data storage parameters. According to Aoyama and Sakai [2011], in a federation of clouds, infrastructure-level resource monitoring data can be collected about the usage status and dead or alive status of computing, storage, and network resources of a cloud system. This monitored data is required to determine the need for load distribution or even disaster recovery. On the application level, the monitored parameters and the way their values should be retrieved depend on the application and not on the cloud infrastructure it is running on. In an integrated cloud environment like a cloud federation, a general monitoring infrastructure is required to collect and process the information provided by the monitored components regardless of the level of parameters being monitored [Rak et al. 2011].

Existing monitoring systems such as Ganglia [Massie et al. 2004], Nimsoft,³⁰ Nagios,³¹ and GridICE [Andreozzi et al. 2005] addressed monitoring of large distributed systems, but the rapidly changing and dynamic nature of services in clouds cannot be addressed thoroughly by these systems. In a federated cloud environment, monitoring is a more complicated task because of the diversity of the clouds and different domains they exist in. Resources may reside across different cloud infrastructures, so the monitoring system must collect and aggregate data from heterogeneous cloud environments. As a result, standardized interfaces and formats are required to enable the federation monitoring.

In such a federated environment, when virtual resources are migrated from one site to a remote site, the monitoring data from the remote site still needs to be collected by the service manager in the origin destination. Moreover, by migrating to a remote site, the originating site loses direct control of the virtual resource and underlying hardware. In order to ensure a continuous monitoring capability, each of the clouds needs federation components and objects to be created and managed to support remote monitoring [Clayman et al. 2010].

As cloud applications get larger and larger and are scattered across clouds, the need for an autonomic monitoring framework that works without intervention and reconfiguration arises. Monitoring tools must be designed to support autonomic federated cloud monitoring. In addition to basic monitoring, in a large environment such as Inter-cloud, the monitoring system requires mechanisms that allow a service to receive messages when events occur in other services and applications. Services interested in receiving such messages are often unknown in advance or will change over time. Therefore, services must be able to register (subscribe) interest for receiving events from the event sources. Experiences with WS-Eventing³² can be helpful in this regard.

Rak et al. [2011] present monitoring components that facilitate the development of custom monitoring systems for cloud applications using the mOSAIC project [Petcu et al. 2011] APIs. Clayman et al. [2010] present the Lattice³³ framework, which has been specially designed for monitoring resources and services in virtualized environments such as the RESERVOIR project [Rochwerger et al. 2009].

5.6. Economy

5.6.1. Market. Interoperability between different providers allows cloud customers to use the service across clouds to improve scalability and reliability [Mihailescu and Teo 2010c]. Computing as a utility can be considered as one of the main goals in federated cloud computing where resources in multiple cloud platforms are integrated in a single resource pool. A key challenge in this regard is how cloud providers interact with each other to realize collaboration [Zhang and Zhang 2012].

A cloud provider is able to meet the peak in resource requirements by buying resources from other cloud providers. Similarly, when a cloud provider has idle resources, it can sell these resources to the federated cloud market. In order to enable such resource sharing and collaboration among cloud providers, there is a need for a marketplace with exchange facilities that helps providers in trading resources among each other [Toosi et al. 2011].

Buyya et al. [2010] propose a federated network of clouds mediated by a cloud exchange as a market maker to bring together cloud providers and customers. It supports trading of cloud services based on competitive economic models such as commodity

³⁰Nimsoft, <http://www.nimsoft.com/index.html>.

³¹Nagios, <http://www.nagios.org/>.

³²Web Service Eventing (WS-Eventing), <http://www.w3.org/Submission/WS-Eventing/>.

³³Lattice framework, <http://clayfour.ee.ucl.ac.uk/lattice/>.

markets and auctions. Federated cloud providers require a clear understanding of the ramifications of each decision they make regarding selling/buying resources to/from other providers. Goiri et al. [2011] present a plausible characterization of providers decisions operating in a federated cloud including outsourcing requests or renting idle resources to other providers.

Market-based approaches for allocation of shared resources have proven their potential in computational systems [Mihailescu and Teo 2010a]. To address the market-based resource allocation mechanism design problem, Mihailescu and Teo [2010b] propose a reverse auction-based mechanism. The market maker selects the sellers for allocation, based on the published price, such that the underlying resource costs are minimized. Afterward, the actual payments for the winning sellers are determined based on the market supply.

5.6.2. Pricing. Pricing and profit are two important factors for cloud providers to remain in the business [Toosi et al. 2011]. Cloud federation allows providers to trade their resources under federation regulations. Strategies regarding selling and buying of resources in federated cloud environments are important issues that should be considered by providers. How providers price their services in the federated cloud market requires profound considerations to ensure the profitability of the providers. To be precise, it is important that the provider has a clear understanding of the potential of each federation decision, and providers should answer questions such as to what extent they want to contribute to the federation or how much they should charge other providers for their service.

Goiri et al. [2011] present a profit-driven policy for decisions related to outsourcing or selling idle resources. They characterize these decisions as a function of several parameters and implement a federated provider that uses this characterization to exploit federation. Toosi et al. [2011] propose similar policies and a way to price resources in the federated cloud. Furthermore, they proposed a financial option-based cloud resource pricing model to help providers in the management of reserved resources [Toosi et al. 2012].

Dynamic resource pricing is a necessity in interconnected cloud environments where distributed cloud providers seek to accommodate more customers while they compete with each other. Mihailescu and Teo [2010b] argue that dynamic pricing is more suitable for federated sharing of computing resources, where participants may both provide and use resources. They present an auction framework that uses dynamic pricing to allocate shared resources. They show that using their proposed dynamic pricing scheme, the user welfare, the percentage of accepted requests, and the percentage of allocated resources increase in comparison to fixed pricing.

5.6.3. Accounting and Billing. In a federated cloud environment, accounting and billing must be carried out in a way that meets the requirements of the cross-cloud scenario. Some identified challenges may affect the design of the accounting and billing in this environment; the actual placement of the resources may not be known to the entire system and may also change during the service lifetime. Moreover, the number of required resources composing a service can dynamically go up and down to cope with a change in demand [Elmroth et al. 2009]. Primarily, it is required that resource usage be monitored for billing and accounting purposes. Additionally, in federated cloud environments, cloud providers expect the federation to be honest in its accounting and billing practices [Harsh et al. 2011].

Any accounting and billing approach must be performed in a fair and standardized way both (a) between cloud customers and Cloud provider and (b) between cloud providers [Elmroth et al. 2009]. Moreover, for billing, those approaches must take into account the postpaid and prepaid payment schemes for capacity that varies over time

in response to customer requirements. Elmroth et al. [2009] present a solution for accounting and billing in a federated cloud environment. The focus of the work is in the design of the accounting and billing system, utilizing existing alternatives, for the RESERVOIR [Rochwerger et al. 2009] project. They focused on accounting and billing between a cloud provider and consumer. However, Inter-cloud accounting and billing still remain as an open issue and need further considerations.

5.7. Network

5.7.1. Network Virtualization and Connectivity. Network connectivity over distributed resources, for example, deployment of a “virtual cluster” spanning resources in different providers, is a challenging issue for both users and providers. Providers are not willing to give users privileged access to the core network equipments because of security risks. Furthermore, creating APIs that reconfigure the network infrastructure based on the user requirements is also difficult. Consequently, creation of a trusted network environment faces challenges in terms of connectivity, performance, and management [Keahey et al. 2009].

One common method of addressing the connectivity problem involving resources in multiple sites is creation of a *virtual network* based on *network virtualization* technologies. A virtual network is an overlay network that consists of virtual resources, such as virtual network interface cards, rather than physical resources and is implemented using methods of network virtualization. The concept of network virtualization has appeared in the networking literature in the past and can be applied to the interconnected cloud environment.

EUCALYPTUS [Nurmi et al. 2009], an open-source cloud platform, provides support for VLANs across multiphysical hosts and requires the VLANs capable managed switch. To support applications that are required to be deployed and migrated across clouds, RESERVOIR [Rochwerger et al. 2009] employs an overlay network between hypervisors. These overlays are called virtual application networks (VANs). Keahey et al. [2009] propose ViNe for Sky Computing that offers end-to-end connectivity among nodes on the overlay network even if nodes are in private networks or protected by firewalls. ViNe supports multiple, mutually isolated virtual networks, which providers can dynamically configure and manage. Regarding this issue, Nagin et al. [2011] propose proxy servers at the source and destination clouds that communicate with each other while hiding the details of the source and destination hosts. VNET [Sundararaj et al. 2004] is a virtual private network that implements a virtual local area network spread over a wide area using layer 2 tunneling. VNET is an adaptive overlay network for virtual machines and is not designed for specific applications.

5.7.2. Addressing. One main challenge in the implementation of long-distance VM migration is the addressing issue. In a virtualized environment like cloud, in the near future the IPv4 address space will not be sufficient as millions of VMs will be running and each one could have a handful of IP addresses associated with it. Therefore, many cloud operators are considering switching to IPv6, which provides much larger local address space. The fact that some cloud builders will use IPv4 and some will use IPv6 is not far-fetched. As a result, a common IP mobility scheme between these two is required [Bernstein et al. 2009].

When a running VM migrates from one location to another, the IP address goes with the running VM and any application hosted by that VM. Both *location* and *identity* are embodied by IP addresses. That is, routers and switches of the network not only identify the endpoint but also infer the location of the endpoint from the IP address. In federated cloud environments, where VMs migrate between geographically distributed

sites, mobility of IP addresses is a challenge. Nevertheless, mobile IP mechanisms³⁴ can be used in this case. However, they are not common between IPv4 and IPv6. The new scheme is called Location Identity Separation Protocol (LISP) and has been proposed by Bernstein et al. [2009] to operate with both IPv4- and IPv6-based networks. LISP facilitates the IP mobility by decoupling location and identity. In summary, any addressing scheme should consider the mobility aspects of VMs in federated cloud environments [Bernstein et al. 2009].

5.7.3. Naming. In federated cloud environments, services, workloads, and applications are distributed across multiple locations and those locations may change on a frequent basis. Finding those services and scaling the rate of change effectively need an efficient dynamic cloud naming system. The Domain Name System (DNS) is designed to locate and address hosts, services, or any resource connected to the Internet. But the flat name space and a simple name lookup that DNS represents is not sufficient for cloud computing. In fact, clouds are not endpoints in the way servers or clients on the Internet are [Bernstein et al. 2009]. Cloud computing environments are endlessly changing environments. In order to enable effective identification of the required service, its capabilities, and required features, audit capabilities are required in the design of a cloud naming system [Bernstein et al. 2009]. Moreover, in federated environments, a cloud naming system should be able to manage frequent name alteration and name space integration. A cloud entity being part of a virtual cloud application could later become part of another cloud application [Núñez et al. 2011].

Clouds include many entities that need to be identified [Celesti et al. 2010d]. In order to enable cloud platforms to manage and control their resources, they need to name, identify, and locate them [Celesti et al. 2010d]. The nature of the resources involved in the cloud computing paradigm varies from physical components (servers, storage units, etc.) to abstract elements (virtual machines, data repositories, applications, etc.). All these real or abstracted entities are offered to users and can be seen as entities of the cloud [Núñez et al. 2011]. In an Inter-cloud scenario, clouds themselves could be seen as potential resources to be exploited, in the form of a high-level component capable of offering computation, storage, and networking.

5.7.4. Multicasting. Cloud computing is a suitable platform for applications with a large number of users and data like multimedia-enabled applications such as Facebook and MySpace. Multicasting is effectively exploited in these massive-scale, real-time, multipoint applications. Cloud providers mostly do not allow IP multicasting³⁵ on their networks, as it imposes a high load on their routers. However, it is a crucial element for the aforementioned applications where multicasting must be supported for their implementation. More significantly, for these types of applications to work in the Inter-cloud context, IP multicast between clouds must be supported. Consequently, interdomain multicasting is required [Bernstein et al. 2009]. This becomes further complicated if a location-agnostic addressing scheme has been adopted, as discussed earlier. Cisco has been actively involved in work in this area.³⁶

5.8. Autonomics

With the growing complexity of interconnected systems such as Inter-cloud, system management duties become too complex to be carried out only with human intervention

³⁴IP Mobility Support for IPv4, revised, at <http://www.ietf.org/rfc/rfc3344.txt>, IP Mobility Support in IPv6, at <http://www.ietf.org/rfc/rfc3775.txt>.

³⁵IP multicast is a method of sending Internet Protocol (IP) datagrams to a group of interested receivers in a single transmission.

³⁶LISP for Multicast Environments, <http://tools.ietf.org/html/draft-farinacci-lisp-multicast-01>.

and manual administration. Hence, to overcome the issue, the need for *autonomic computing* becomes more and more tangible. Autonomic computing refers to the self-managing principles of computer-based systems while hiding the intrinsic complexity of the system. Using the holistic techniques provided by autonomic computing, we can handle to a large extent different system requirements such as performance, fault tolerance, reliability, security, QoS, and so forth without manual intervention.

In heterogeneous and dynamic interconnected cloud environments, the system must continuously adapt itself to the current state of the system. The result must be an integrated solution capable of a wide range of autonomic management tasks including *self-configuration* (i.e., automatic configuration of components), *self-healing* (i.e., automatic discovery and correction of faults), *self-optimization* (i.e., automatic optimization of resource allocation), and *self-protecting* (i.e., automatic system security and integrity). Self-management of cloud services minimizes user interactions with the system and represents challenging research issues. There is big overlap between autonomics and other challenges and related issues we discussed in this article. That is, autonomic systems can be utilized for different aspects of interconnected cloud environments such as SLA management, provisioning, security, market, and so forth. This requires a detailed investigation of autonomic computing for each aspect.

For example, autonomic computing principles can be applied for provisioning. This is because in interconnected cloud environments, user applications might require expanding their resources by scaling out onto another cloud, and may have preference for a particular cloud or may want to combine multiple clouds. Such integration and interoperability must be done without manual intervention. In addition, in a federated cloud environment, a small or private cloud might be required to expand its capacity or computational resources by integrating or bursting into other cloud platforms on demand. Such dynamic and scalable provisioning must be done autonomously based on the workload, spikes in demands, and other extreme requirements.

Self-manageable interconnected cloud infrastructures on one hand are required to achieve a high level of flexibility and on the other hand to comply with users' requirements specified by SLAs. Matching desired user SLAs with cloud providers' service offerings is a challenge. That is, due to a large variety of services and offerings in cloud environments, matching between user requirements and services is a difficult task. Flexible and adaptive SLA attainment strategies are needed to overcome this issue. Such flexible and dynamic SLAs cannot be generated manually due to the high number of services and consumers and requires to be done autonomously.

Relevant progress in the field of autonomic cloud computing has been achieved by Kim and Parashar in the CometCloud project [Kim and Parashar 2011]. CometCloud is an autonomic computing engine developed for cloud and grid environments that supports highly heterogeneous infrastructures, integration of public and private clouds, and dynamic application scale-out. The service layer of CometCloud provides a range of services to support autonomics at the programming and application levels, including features such as deadline-, budget-, and workflow-based autonomic scheduling of applications on the cloud, fault tolerance, and load balance.

Other approaches apply autonomic computing principles for specific aspects of cloud computing such as market management [Breskovic et al. 2011], cloud networking [Choi et al. 2011], VM configuration [Xu et al. 2012], and workflow execution [Papuzzo and Spezzano 2011].

6. STANDARDS

Successful realization of Inter-cloud requires different standards. In particular, standards for interoperability, security, and legal issues must be taken into account when a platform for interoperability among different cloud vendors is created. Stepping toward

a commonly and widely adopted solution requires a considerable amount of work and research to overcome existing diversities. There are several barriers and problems in applying standard APIs and protocols in cloud environments [Petcu 2011]:

- (1) Vendors usually prefer to lock in their customers with their facilities to avoid losing them to competitors.
- (2) Cloud providers offer differentiated services and desire to have their own particular services to attract more customers.
- (3) Cloud providers often do not easily agree on certain standards.
- (4) It takes years to fully develop a standard and apply it globally.
- (5) There are numerous standards being developed simultaneously, and agreement on which one to adopt may be difficult and sometimes impossible to attain.
- (6) In cloud computing, substantially different standards are required for diverse cloud models (e.g., IaaS, PaaS, SaaS). Accordingly, one comprehensive set of standards is hard to develop.

There are many groups and initiatives that are working on cloud computing standards. We identified the main active groups and their activities and summarized them in Table I. These groups and organizations can be categorized into two main categories:

- Standards developing organization (SDO)*, when they are technically involved in developing and publishing standards for cloud computing and cloud interoperability; and
- Industrial or scientific consortia and standards-setting organization (SSO)*, when they work toward promoting the adoption of emerging technologies, typically without the intention of developing their own standards. Consortia bring organizations, companies, academia, and governmental institutes together to cooperate toward the purpose of wider adoption and development of cloud computing technologies and they are interested in achieving a consensus to address technical problems in cloud computing. The efforts of consortia and SSOs expedite the standard development process or even in the case of being widely accepted by cloud computing stakeholders are converted to standards.

Bold entries in Table I represent SDOs. A more detailed review of the current standard protocols and interfaces facilitating Inter-cloud realization can be found in the supplementary Appendix B. Interested readers will also find an inventory of standards relevant to cloud computing³⁷ compiled by the National Institute of Standards and Technologies (NIST) Cloud Computing Standards Roadmap Working Group (CC-SRWG). The Working Group actively updates the inventory as and when more standards are created.

7. INTER-CLOUD PROJECTS

7.1. RESERVOIR

The RESERVOIR [Rochwerger et al. 2009] project introduces modular, extensible, and open cloud architecture that supports business-driven cloud federation. It enables cloud infrastructure providers, from different administrative domains, to collaborate with each other in order to create a vast pool of resources while technological and business management decisions on each domain are made autonomously by the provider.

In the RESERVOIR model, service and infrastructure providers play different functional roles. Service providers offer service to their customers' applications based on

³⁷Inventory of Standards Relevant to Cloud Computing, <http://collaborate.nist.gov/twiki-cloud-computing/bin/view/CloudComputing/StandardsInventory>.

Table I. Standardization Activities Regarding Inter-Cloud Challenges: Provisioning (Pr), Portability (Po), Service-Level Agreement (SLA), Security (S), Monitoring (M), Economy (E), Network (N), Autonomics (A)
Bold entries are standards developing organizations.

	Pr	Po	SLA	S	M	E	N	A
DMTF^a	✓	✓	✓	✓	✓			✓
OGF^b	✓		✓	✓	✓			✓
CSA^c				✓	✓			
OCM^d		✓	✓					
NIST^e	✓	✓		✓				
CCIF^f	✓			✓	✓			
OCC^g							✓	
OASIS^h	✓		✓	✓				
GICTFⁱ							✓	✓
ETSI^j		✓					✓	✓
CWG^k						✓		
OMG^l			✓	✓		✓		
ODCA^m	✓		✓	✓				
IEEE P2302ⁿ	✓	✓		✓	✓	✓	✓	
SNIA CSI^o	✓	✓						
ISO JTC 1/SC 38^p	✓	✓	✓		✓	✓	✓	
ITU-T FG^q	✓	✓	✓	✓	✓		✓	
SIENA^r	✓	✓		✓				✓

^aDistributed Management Task Force (DMTF), <http://www.dmtf.org/>.

^bOpen Grid Forum (OGF), <http://www.gridforum.org/>.

^cCloud Security Alliance (CSA), <https://cloudsecurityalliance.org/>.

^dOpen Cloud Manifesto, <http://www.opencloudmanifesto.org/>.

^eNational Institute of Standards and Technologies (NIST), <http://www.nist.gov/>.

^fCloud Computing Interoperability Forum (CCIF), <http://www.cloudforum.org/>.

^gOpen Cloud Consortium (OCC), <http://opencloudconsortium.org/>.

^hOrganization for the Advancement of Structured Information Standards (OASIS), <https://www.oasis-open.org/>.

ⁱInterCloud Technology Forum (GICTF), http://www.gictf.jp/index_e.html.

^jThe European Telecommunications Standards Institute (ETSI), <http://www.etsi.org/>.

^kThe Open Group Cloud Computing Work Group, <http://www.opengroup.org/getinvolved/workgroups/cloud-computing>.

^lObject Management Group (OMG), <http://www.omg.org/>.

^mOpen Data Center Alliance (ODCA), <http://www.opendatacenteralliance.org/>.

ⁿIEEE P2302 Working Group (Intercloud), <http://grouper.ieee.org/groups/2302/>.

^oStorage Networking Industry Association (SNIA) Cloud Storage Initiative, <http://www.snia.org/forums/csi>.

^pISO JTC 1/SC 38, http://www.iso.org/iso/home/standards_development/list_of_iso_technical_committees/jtc1_home/jtc1_sc38_home.htm.

^qITU-T Focus Group on Cloud Computing (FG Cloud), <http://www.itu.int/en/ITU-T/focusgroups/cloud/>.

^rStandards and Interoperability for eInfrastructure implementation initiative (SIENA), <http://www.sienainitiative.eu/>.

leased resources of infrastructure providers. Infrastructure providers provide a seemingly infinite pool of virtualized computational, network, and storage resources. These virtualized resources are offered in the form of fully isolated runtime environments called virtual execution environments (VEEs). VEEs abstract away the physical characteristics of the resources and enable resource sharing.

Every RESERVOIR site includes three different abstract layers: Service Manager, Virtual Execution Environment Manager (VEEM), and Virtual Execution Environment Host (VEEH). The Service Manager, the highest level, receives a service manifest from the service provider. Service Manager handles several tasks such as deploying and provisioning VEEs, billing, accounting, and monitoring SLA compliance. VEEM, the second layer, is responsible for managing VEEs and interacting with VEEM on remote sites allowing federation of infrastructures. VEEM is also responsible for optimal placement of VEEs into VEE hosts according to constraints determined by Service

Manager. The lowest level, VEEH, supports different virtualization platforms for control and monitoring of VEEs. Moreover, transparent VEE migration within the federated cloud is supported by VEEH.

7.2. mOSAIC

mOSAIC [Petcu et al. 2011] is a multicloud solution for cloud application developers to help them to see the cloud resources as abstract building blocks in their application. It deals with the cloud issues by focusing on the application layer rather than the IaaS layer. mOSAIC enables application developers to obtain the desired application characteristics such as scalability, fault tolerance, and QoS.

One of the main goals of mOSAIC is to allow transparent and simple access to heterogeneous cloud resources and to avoid vendor lock-in. It fulfills this goal by its cloud ontology that describes services and their interfaces. Moreover, a unified cross-platform API that is platform and language independent is provided by mOSAIC.

The mOSAIC platform is targeted mainly toward cloud application developers. Therefore, mOSAIC intends to offer them an SLA-oriented resource management based on agent technologies. Inside its platform, several different agents are provided to support resource-related services such as resource discovery, negotiation, brokering, monitoring, tuning, and scalability. It also provides an event-driven approach to adapt the cloud configuration according to changes in application requirements. All these capabilities establish a framework for dynamically interconnecting and provisioning services from multiple clouds.

7.3. Contrail

The Contrail project [Carlini et al. 2012] proposes a federated and integrated cloud approach. Contrail tries to create an environment that allows cloud customers to exploit resources belonging to different cloud providers through a homogeneous secure interface regardless of the technology the providers use. In addition, it promotes adoption of a fully open-source approach toward this goal.

Contrail integration can be categorized in two parts: vertical and horizontal integration. In the vertical integration, a unified platform for accessing different resources is provided, while in the horizontal integration, the interaction between different cloud providers has been provided. Contrail works based on the broker services (*federation support*) that act as mediators between cloud users and providers. The federation support offers resources belonging to different cloud providers to users in a uniform fashion.

The federation architecture is composed of three layers, namely, *interface*, *core*, and *adapters*. The interface layer gathers requests from users as well as other Contrail components that rely on the federation functionality and facilities. The interface layer includes a Command-line interface and a web interface, from which it is possible to access REST services. The core layer contains modules for identity management, application deployment, and SLA coordination.

The identity management provides a federation-level account to each user. By using this account, the user can have access to all the resources owned by the federated cloud providers. Single Sign-On (SSO) has been provided by federation support; that is, once a user is authenticated and gains access to the federated cloud providers, the user is not prompted again to log in at each of them.

The Federation Runtime Manager (FRM) component in the core layer is responsible for application deployment. FRM provides discovery and selection to minimize economical costs and to maximize performance levels. Moreover, FRM is responsible for the application life cycle management. The Image manager and provider watcher are two

other components in the core, which are in charge of managing images and monitoring processes, respectively.

One of the main components in the core layer is the SLA Organizer. It extends the SLA management functionalities of the SLA@SOI³⁸ project. The SLA Organizer is a collection of three modules: SLA Coordination, SLA Negotiation, and SLA Template Repository.

The adapters layer contains the internal and external modules that enable access to infrastructural services for both the Contrail cloud and external clouds, respectively. Internal adapters provide components for network (Virtual Infrastructure Network [VIN]), storage (Global Autonomous File System [GAFS]), and computing (Virtual Execution Platform [VEP]). External adapters supply provider-specific adapters for non-Contrail providers by translating requests from the federation support into requests that are understood by the provider.

7.4. Cloudbus InterCloud

InterCloud [Buyya et al. 2010] promotes interconnected cloud computing environments that facilitate scalable provisioning of application services based on QoS constraints under variable workload, resource, and network conditions. It supports scaling of applications across multiple clouds according to required resources for cloud applications (VMs, services, storage, and database) in order to handle sudden variations in service demands.

InterCloud is composed of a set of elements that interact via a market-oriented system to enable trading of cloud resources such as computing power, storage, and execution of applications. The Inter-cloud model comprises two main elements: *Cloud Exchange* and *Cloud Coordinator*.

The *Cloud Exchange* component offers services regarding the information system directory and market making that allow providers to find each other and directly trade cloud resources. In the former case, it implements a web-service-based interface that allows providers to join and leave the federation. In the latter case, with the aim of finding available resources, providers send requests for resources to the Cloud Exchange. Providers who are interested in selling resources publish their offers to the Cloud Exchange as well. The Cloud Exchange generates a list of providers with corresponding service prices that can handle requests according to the market mechanism. In this way, buyers are able to locate potential sellers for the required resources.

The *Cloud Coordinator* component is responsible for domain- \hat{A} -specific issues related to the federation. Every provider in the federation contains this component. The Cloud Coordinator has two main parts: front-end, which is responsible for interaction with the federation, and back-end, which interacts with the associated provider. Front-end components interact with the Cloud Exchange and other coordinators. The former allows data centers to publish their offers and requests for resources, whereas the latter allows the Coordinator to acquire the current state of the provider to decide about allocation of additional resources from the federation or the amount of offering resources with other members. Hence, wherever the Coordinator detects that additional resources are required, it sends requests to the federation to discover potential seller providers. Once potential providers are discovered and the preferred one is selected, the Coordinator contacts the remote Coordinator and they start the resource exchange process. Similarly, when the Cloud Coordinator notices that local resources are underutilized, it can publish an offer for idle resources in the Cloud Exchange in order to find potential buyers.

³⁸SLA@SOI, <http://sla-at-soi.eu/>.

Inter-cloud acts at high levels of cloud interoperability, and issues related to security, VM images, and networking are not handled by the framework. However, existing approaches or even new solutions for them can be applied in Inter-cloud without modifying its architecture.

7.5. OPTIMIS

OPTIMIS [Ferrer et al. 2012] is a toolkit that enables flexible and dynamic provisioning of cloud services targeting multicloud architectures. The focus of the toolkit is on cloud service and infrastructure optimization throughout the construction, deployment, and operation phases of the service life cycle. It provides a platform for consumers to employ cloud services with requirements regarding allocation of data and VMs such as elasticity, energy consumption, risk, cost, and trust. In terms of provisioning models for cloud computing, OPTIMIS facilitates cloud bursting, multicloud provisioning, and federation of clouds. Multicloud architectures and federated cloud environment in OPTIMIS enable transparent, interoperable, and an architecture-independent fashion of utilizing resources from multiple providers.

The toolkit consists of a set of fundamental components in order to realize different multiple cloud scenarios. The main components of the toolkit are the Service Builder, the Basic Toolkit, the Admission Controller, the Deployment Engine, the Service Optimizer, and the Cloud Optimizer. *Service Builder* allows a service programmer to access an integrated development environment. It simplifies both the development and configuration of the service using a novel programming model for service development. The *Basic Toolkit* provides functionalities common to components that are used during service deployment and execution (e.g., monitoring and security).

Providers receiving a deployment request perform an admission control to decide whether to admit the request. Using the Basic Toolkit, in order to choose the most suitable provider, the *Deployment Engine* evaluates the providers' offers to run the service. Afterward, allocation of resources for the service is performed by the *Cloud Optimizer* with help from components for management of VMs and data using functionalities in the Basic Toolkit. The *Service Optimizer* is notified once the deployment process is completed. According to the agreed-upon SLAs, the *Service Optimizer* continuously checks the service, and if it is required, it can migrate the service to another provider.

7.6. Open Cirrus

Open Cirrus [Avetisyan et al. 2010] is a federation-based cloud computing testbed sponsored by companies such as HP, Yahoo!, and Intel and supported by academic institutions in the United States, Germany, and Singapore. It is composed of data centers located in the United States, Europe, and Asia. It offers researchers access to the federated infrastructure. Each data center (site) is organized as a stack of services.

At the lowest layer, referred to as the foundation layer, a service called *Zoni* is offered. Zoni offers services at the IaaS layer and is responsible for activities such as management of physical resources and services such as resource allocation, node isolation, software provisioning, logging, and networking. The next layer, called *primary domain services*, offers services at the PaaS level. It enables users to use application frameworks such as Hadoop and to execute MPI applications without having to interact with the foundation layer. This layer also offers cloud storage (via the Hadoop File System) and management of virtual machines deployed in multiple sites via an AWS-compatible API. Finally, the *utility services* offer additional non-critical services such as monitoring, network file system, and accounting for resource utilization.

Allocation of resources belonging to multiple sites is a user-initiated task. It is facilitated via services such as a global sign-on,³⁹ global monitoring tools, and user directories remotely mountable, and global storage. It is worth noting that, even though allocation of resources at the federation level is a user-initiated task, the final decision on allocation is made by individual remote sites. Therefore, it is possible that a particular user will not have access to a particular site of the federation. As users are assigned to a particular site, utilization of remote resources (i.e., resources that belong to other sites rather than the one the user belongs to) incur credits being transferred from the local to the remote site for compensation for resource utilization.

7.7. Claudia

Claudia [Rodero-Merino et al. 2010] is a user-side system for cloud abstraction that aims at providing a single interface for cloud service management and auto-scaling abilities for cloud service providers (the intended users of the system). Claudia enables different services to be deployed independently via service description files (SDFs). This enables service providers to describe service dependencies, which are handled by Claudia adequately: dependencies are identified and deployed before dependent services are deployed. Claudia's SDF language is an extension of the OVF.

An important feature of Claudia is the ability to perform automated auto-scaling based on user-defined scalability rules. Such rules define the trigger for the auto-scaling feature (either service metrics such as response time or hardware metrics such as CPU utilization) and the scalability action to be carried out by the system. When the trigger conditions are detected, the action is automatically performed by the system. Scalability rules encompass both scaling resources up/down (i.e., increasing or decreasing the number of resources from a specific provider) and scaling resources in/out (i.e., consolidating or spreading resources across different providers).

Access to the underlying cloud infrastructure is performed by independent cloud infrastructure managers that must be able to interact with multiple cloud providers. In this sense, client-side libraries, described in Section 8.2, could be adopted for this purpose. Alternatively, systems such as OpenNebula [Moreno-Vozmediano et al. 2012], which are also able to interact with multiple clouds, can also be used for this purpose. Claudia is developed as part of the RESERVOIR [Rochwerger et al. 2009] project.

7.8. Intercloud by Bernstein et al.

Bernstein et al. [2009] propose a blueprint for interconnection of cloud data centers. The blueprint focuses on challenges in low levels, such as virtual machine mobility and interoperability, storage interoperability, network addressing, and addressing mobility, security (mainly identity and trust), and messaging. In this direction, this research has been targeting protocols and mechanisms for Inter-cloud. Advanced features not addressed by other initiatives but considered in the blueprint are multicasting, time synchronization, VM formats, reliable transport, events sequencing, and storage replication [Bernstein and Vij 2010c].

The establishment of Intercloud starts with elements called *Intercloud Root Instances* [Bernstein et al. 2009] that are responsible for mediating connection between different cloud providers. This avoids the necessity of each provider having to mediate the access to other providers in Intercloud, which helps in increasing the system scalability.

Besides InterCloud Root Instances, the architecture also contains *Intercloud Exchange Providers* that are responsible for aiding with the negotiation between providers for utilization of resources. Finally, a Catalog component stores information necessary

³⁹Global sign-on enables users to access any federated site via a single set of credentials.

for providers to locate other members of the Intercloud and the offered services. Catalogs are locally made available by providers, and the Exchange is able to aggregate information from these multiple catalogs in order to handle complex queries from other members.

Most of the work regarding the blueprint is directed toward concepts, architectures, and standards rather than actual system developments. It also includes protocols necessary for enabling different parts of the Intercloud interactions [Bernstein et al. 2009].

7.9. Federated Cloud Management (FCM)

Federated Cloud Management (FCM) [Marosi et al. 2011] is an architecture that enables the integration of multiple IaaSes to execute applications from multiple users. It is composed of a number of elements. The core element of the architecture is the *Generic Meta Brokering Service* (GMBS) component that receives requests from multiple users and can direct them to multiple IaaS clouds for execution. The GMBS performs the matchmaking between user requests and resources from clouds to determine where the request should be scheduled.

A specific *Cloud-Broker* for each available IaaS provider interacts with the GMBS to receive the user request and to execute it in its infrastructure. IaaS brokers are also responsible for managing resources and keep QoS and resource utilization metrics updated so it is used by the GMBS during the matchmaking process. Virtual appliances that contain the necessary software to support user requests are kept in a virtual appliance repository that is part of the FCM architecture. These virtual appliances are reconstructed on each cloud provider, when required by user requests, by a *VM Handler* component that is executed on each IaaS cloud.

7.10. Sky Computing

The Sky Computing project [Keahey et al. 2009] aims at enabling aggregation of multiple virtualized sites in order to enhance availability of resources. This project addresses issues such as trust, VM portability, and connectivity of geographically spread resources. The latter is achieved via overlay networks that enable remote resources to access each other as if they were connected in a local area network. The different features from the project that enable interoperability are achieved via utilization of middleware. Demonstrations of the feasibility of the approach were performed with leverage of different existing open-source tools such as Xen, ViNE (for overlay networking), Nimbus, and Hadoop.

7.11. STRATOS

STRATOS [Pawluk et al. 2012] proposes a broker enabling allocation of cloud resources from multiple providers. The decision of the providers to be utilized for a specific allocation is defined at runtime. The proposed architecture performs selection and monitoring and is able to consider SLA for decision making. Among the use cases proposed by the authors, the goals of the broker were avoiding lock-in and minimizing deployment cost. One of the main differences of this work in relation to competing approaches is the consideration of service measurements and KPIs for the research selection via the SMI (Service Measurement Index) framework.⁴⁰

8. TOOLS AND FRAMEWORKS

Apart from the Inter-cloud projects we discussed in the previous section, there are tools and frameworks that play a crucial role in enabling cloud interoperability. In

⁴⁰<http://www.cloudcommons.com/about-smi>.

this section, we initially discuss open-source cloud platforms that facilitate cloud interoperability and portability. Then, we discuss client-side libraries and distributed programming languages that provide abstract programming facilities for clients to build their own multicloud solutions. Finally, we discuss projects and tools providing interconnected cloud platforms for scientific applications and research purposes.

8.1. Open-Source Cloud Management Platforms

Open-source cloud platforms (OCPs) are important for cloud interoperability not only because of the benefits of being open source but also because they are able to mitigate the risk of vendor lock-in by providing interoperable cloud environments. As OCPs mostly support standard interfaces such as Open Grid Forum (OGF)⁴¹ and Open Cloud Computing Interface (OCCI),⁴² applications deployed on OCPs can be easily moved from one IaaS provider to another one implementing these APIs, without having to be modified. Considering the fact that OCPs facilitate cloud interoperability and portability, we study them among Inter-cloud solutions. We briefly explore four main OCPs and their architectures: OpenNebula⁴³ [Moreno-Vozmediano et al. 2012], OpenStack,⁴⁴ CloudStack,⁴⁵ and Eucalyptus⁴⁶ [Nurmi et al. 2009].

8.1.1. OpenNebula. OpenNebula [Moreno-Vozmediano et al. 2012] is an open-source platform for management of virtualized data centers to enable IaaS clouds. OpenNebula's main application is as a tool to manage a virtualized infrastructure in private, public, or hybrid clouds. OpenNebula is not only designed for cloud interoperability but also for comprehensive management of virtualized data centers. Interoperability and portability, leveraging and implementing standards, adaptability to manage any hardware and software, and scalability of large-scale infrastructures are among the main principles considered in the design of OpenNebula.

The OpenNebula architecture consists of three layers:

- Tools:** Contains OpenNebula's command line interface (CLI), the scheduler and interfaces for communication with the *Core* layer. The scheduler is an independent entity that uses an XML-RPC interface to invoke actions on virtual machines. The *Haizea* lease manager [Sotomayor et al. 2009] can also be used as a scheduling module in OpenNebula. Haizea allows OpenNebula to lease resources as VMs, with a variety of lease terms supported, including advance reservation of resources and best-effort requests.
- Core:** Consists of components to control and monitor virtual machines, virtual networks, storage, and hosts. The core layer performs its actions by invoking a suitable driver.
- Drivers:** Contains drivers for virtualization, storage, monitoring, and authorization and connects to the underlying physical infrastructure.

OpenNebula provides a higher level of interoperability for private clouds by supporting the most common hypervisors, such as KVM, VMware, and Xen and its libvirt plug-in. In the public cloud, interoperability is provided by supporting the most common cloud interfaces, such as VMware vCloud, OCCI, and open libraries, such as libcloud⁴⁷

⁴¹Open Grid Forum, <http://www.ogf.org/>.

⁴²Open Cloud Computing Interface, <http://occi-wg.org/>.

⁴³OpenNebula, <http://opennebula.org/>.

⁴⁴OpenStack, <http://www.openstack.org/>.

⁴⁵CloudStack, <http://Cloudstack.apache.org/>.

⁴⁶Eucalyptus, <http://www.eucalyptus.com/>.

⁴⁷libcloud, <http://libcloud.apache.org/>.

and δ -Cloud.⁴⁸ Interoperability and portability in the hybrid cloud are also enabled by supporting the combination of local private infrastructure with Amazon EC2 and ElasticHosts, Rackspace, GoGrid, or Terremark through the RedHat's δ -Cloud APIs.

8.1.2. OpenStack. OpenStack is an open-source IaaS Cloud management platform, released under the terms of the Apache License, designed to control large pools of compute, storage, and networking resources in a data center. OpenStack is not specifically designed for either interoperability or portability; nevertheless, it is very close to being a standard in the cloud ecosystem.

OpenStack provides a web interface (dashboard) and Amazon EC2-compatible APIs that can be used by users to provision resources. Similar to OpenNebula, OpenStack also supports OCCI. Since OpenStack APIs are compatible with Amazon EC2 and Amazon S3, applications designed for Amazon Web Services can be used with OpenStack with minimal modification effort.

To maximize interoperability and deployment flexibility and to reduce risks of lock-in associated with proprietary platforms, OpenStack is designed as a series of loosely coupled components that are easy to integrate with a variety of solutions and hardware platforms. The main components are:

- OpenStack Compute (Nova)*: It manages the life cycle of VM instances from scheduling and resource provisioning to live migration and security rules.
- OpenStack Storage (Swift)*: Swift is a scalable redundant storage system responsible for enabling data replication and ensuring integrity.
- Block Storage (Cinder)*: The block storage system allows users to create block-level storage devices that can be attached to or detached from VM instances.
- OpenStack Networking (Neutron)*: Neutron is a system for managing networks and IP addresses. The system allows users to create their own networks and assign IP addresses to VM instances.
- OpenStack Dashboard (Horizon)*: It provides users and administrators with management capabilities via a web interface. Management actions enabled by this component include VM image management, VM instance life cycle management, and storage management.
- OpenStack Identity (Keystone)*: Keystone is an account management service that acts as an authentication and access control system.
- OpenStack Image (Glance)*: It supplies a range of VM image management capabilities from discovery and registration to delivery services for disk and server images.

8.1.3. CloudStack. CloudStack is an open-source IaaS platform originally developed by Cloud.com and later purchased by Citrix. The source code was later denoted by Citrix to the Apache Software Foundation and was released under Apache license in April 2012. CloudStack was designed to support the deployment and management of large networks of virtual machines as an IaaS cloud computing platform. CloudStack supports both the Amazon EC2 and vCloud APIs, in addition to its own API. CloudStack has a hierarchical structure that enables management of large networks of virtual machines. The CloudStack structure includes the following components:

- Hosts*: Physical machines onto which virtual machines are provisioned
- Cluster*: A group of physical machines that utilize the same type of hypervisor
- Pod*: A rack in a data center containing one or more clusters and a switch shared by all clusters in that pod
- Zone*: Collection of pods and secondary storage shared by all pods in the zone

⁴⁸ δ -Cloud, <http://deltacloud.apache.org/>.

Table II. Comparison of Open-Source Cloud Management Platforms

	OpenNebula	OpenStack	CloudStack	Eucalyptus
License	Apache v2.0	Apache v2.0	Apache v2.0	GPL v3
API Compatibility	AWS, OCCI	AWS, OCCI	AWS	AWS
Hypervisor Support	Xen, KVM, VMware	Xen, KVM, VMware	KVM, Xen, VMware, Oracle VM	Xen, KVM, VMware
Architecture	Loosely Coupled	Component Based	Tightly Coupled	Tightly Coupled
Hybrid Cloud	Yes	No	Yes	Yes

—*Primary Storage*: Shared storage across a cluster used to host the guest virtual machines

—*Secondary Storage*: Shared storage in a single zone used to store virtual machine templates, ISO images, and snapshots

8.1.4. *Eucalyptus*. Eucalyptus⁴⁹ [Nurm et al. 2009] is an open-source platform, compatible with Amazon Web Services' (AWS) APIs, for building private and hybrid cloud computing environments. Eucalyptus provides a platform for managing pools of compute, storage, and network resources that can be dynamically provisioned based on the application requirements. In order to maintain compatibility, Eucalyptus Systems announced a formal agreement with AWS in March 2012. Eucalyptus enables workload migration and deployment of hybrid cloud environments.

The Eucalyptus platform is composed of the following high-level components:

- Cloud Controller*: Manages the underlying virtualized compute, network, and storage resources and provides an Amazon EC2-compatible web interface and APIs. Moreover, it handles authentication and accounting.
- Cluster Controller*: Communicates with the Storage Controller and Node Controller and manages virtual machines' execution and SLAs
- Storage Controller*: Provides block storage equivalent to AWS Elastic Block Storage (EBS) that can be dynamically attached to VMs
- Node Controller*: Hosts the virtual machine instances and manages the virtual network endpoints during the VM life cycle using the functionality provided by the hypervisor
- Walrus*: Provides persistent storage service compatible with Amazon S3
- VMware Broker*: Is an optional component that offers an AWS-compatible interface for VMware and runs on the Cluster Controller

At present, Amazon⁵⁰ is one of the dominant players in the IaaS cloud market and its service APIs are becoming de facto standards for operation on cloud resources; accordingly, other vendors offer AWS-compatible APIs for their services. Not surprisingly, all open-source platforms we discussed in this section support AWS-compatible APIs, what makes multicloud deployment scenarios further attainable. A comparison of the discussed cloud platforms is presented in Table II.

⁴⁹Eucalyptus, an acronym for “Elastic Utility Computing Architecture for Linking Your Programs to Useful Systems.”

⁵⁰Amazon Web Services, <http://aws.amazon.com/>.

8.2. Client-Side Libraries

Different projects are focusing on the development of libraries that enable clients to build their own Inter-cloud solutions at different abstraction levels. These libraries offer high-level APIs that exempt developers to decide about interaction with specific cloud providers at application development time. It brings extra flexibility to public cloud utilization, because it facilitates both consideration of new cloud providers for application deployment and changes in the types of instances to be used with the application. It also helps in avoiding vendor lock-in by forcing developers to utilize library-defined services rather than utilizing specialized features that are specific to a given provider.

The jcclouds library⁵¹ provides programmers with abstractions representing typical compute elements (called *ComputeService*) and key-value data stores (called *BlobStore*). As the APIs defining these two elements are vendor agnostic, deployment of cloud-ready applications is made independent from the underlying IaaS provider. jcclouds supports development of applications in both Java and Clojure and offers extra features such as ability for unit testing, load balancing, location-aware abstractions, and mappers that eliminate the need for programmers to interact with web services-based APIs. Providers supported by jcclouds include Amazon AWS, GoGrid,⁵² Windows Azure,⁵³ CloudSigma,⁵⁴ and Ninefold,⁵⁵ among others.

LibCloud⁵⁶ is a Python-based library that, like jcclouds, offers abstractions for compute elements, storage, and load balancing. LibCloud also offers an API for interaction with IaaS-provided DNS. This project supports over 26 different cloud providers.

The δ -Cloud⁵⁷ library is a Ruby-based library for interaction with public cloud providers. Its features are similar to those offered by LibCloud and jcclouds and it supports over 15 different cloud providers.

The main difference between the aforementioned projects are the programming language chosen for the library development and the maturity level of each project, whether in terms of cloud providers supported or in terms of cloud services that can be abstracted via the library's API.

8.3. Distributed Programming Languages

Inter-cloud interoperability can be achieved not only from client-side libraries but also with the use of distributed programming languages such as Nomadic Pict [Sewell et al. 2010] and SALSA [Varela and Agha 2001]. These programming languages allow the algorithms or applications to be executed independent of their locations and transparent to migrations. For instance, SALSA, an actor-based language for Internet computing, provides facilities for applications composed of SALSA actors to be easily reconfigured at runtime by using actor migration. This only requires cloud applications to contain SALSA migratable components and does not impose any further restrictions on workloads. By using this kind of application-level migration, it is possible to achieve load balancing, elasticity, and scalability with finer granularity (in the scale of application entities instead of VM-level coarse granularity). With the same objectives, Imai et al. [2012] proposed a middleware framework to support autonomous workload elasticity

⁵¹jcclouds, <http://code.google.com/p/jcclouds/>.

⁵²GoGrid, <http://www.gogrid.com/>.

⁵³Windows Azure, <http://www.windowsazure.com/>.

⁵⁴<http://www.cloudsigma.com/>.

⁵⁵Ninefold, <http://ninefold.com/>.

⁵⁶LibCloud, <http://libcloud.apache.org/>.

⁵⁷ δ -Cloud, <http://deltacloud.apache.org/>.

and scalability based on application-level migration to cloud computing, also targeting hybrid clouds.

8.4. Interoperable Cloud Infrastructure Projects Supporting e-Science

While all previously discussed projects try to build interconnected cloud environments that are independent of the application type, there are projects that focus on forming distributed multiple cloud infrastructures for scientific applications and for research purposes. In this section, we briefly cover these projects and we categorize them as interoperable cloud infrastructures supporting e-science.

The Open Science Data Cloud (OSDC)⁵⁸ is a distributed cloud-based infrastructure that provides platforms for users to compute over large scientific datasets. The OSDC operates one storage cloud named *Root* and two main compute clouds named *Adler* and *Sullivan*. Adler is an Eucalyptus-based cloud and Sullivan is an OpenStack-based cloud. Root is a repository of various public scientific datasets that can be accessed from the OSDC clouds. The European Grid Initiative is also looking into how to make a grid of academic private clouds and virtualized resources (federate clouds) while focusing on the requirements of the scientific community.⁵⁹ Their goal is to provide an e-infrastructure for research based on the federated operations services.

Aneka [Calheiros et al. 2012b] is a cloud platform that supports creation and deployment of scientific applications across multiple IaaS clouds including a private cloud (one created using Aneka) and public clouds such as Amazon EC2 and Microsoft Azure. Parashar et al. [2013] comprehensively explore benefits, limitations, and research challenges of executing high-performance computing (HPC) scientific workloads across a federation of multiple resources including clouds. Vázquez et al. [2009] present an architecture to build a grid infrastructure with a unified point of access hiring compute resources from public cloud providers with potentially different interfaces to execute HPC applications. They use available technologies rather than developing new standards for future use. The proposed architecture is able to dynamically expand and use resources from cloud providers to react to peak demands. In contrast, Bittencourt et al. [2010] propose an infrastructure able to manage the execution of workflows on a hybrid system composed of both grid and cloud technologies. Vöckler et al. [2011] leverage Pegasus and Condor to execute an astronomy workflow on virtual machine resources provisioned from multiple clouds. Similarly, Gorton et al. present [2010] a federated cloud-based architecture for modeling, simulation, and experimentation of bioinformatics applications.

9. SUMMARY, DISCUSSION, AND FUTURE DIRECTIONS

As the adoption of cloud as the main technology for provisioning of infrastructure, platform, and service for users grows continually, the need to aggregate services and functionalities from different providers arises. This aggregation can happen in any of the delivery models (IaaS, PaaS, or SaaS) and can be enabled by different approaches and technologies.

In this article, we surveyed the relevant aspects that motivate cloud interoperability and the mechanisms and technologies enabling it. We discussed why aspects such as scalability, resource limitations, vendor lock-in, availability, disaster recovery, geographic distribution, latency reduction, regulation and legislation, cost efficiency, and energy savings play a vital role in pushing technologies for cloud interoperability.

⁵⁸Open Science Data Cloud (OSDC), <https://www.opensciencedatacloud.org/>.

⁵⁹European Grid Community, <http://www.egi.eu/infrastructure/cloud/>.

Besides specific motivation, interoperability can be achieved via one or more standard interfaces, brokers, or middlewares. Any of these approaches can be applied to the surveyed interoperability scenarios, which are:

- Federation*, when interoperation is enabled by direct agreements between cloud providers and is transparent to end-users;
- Hybrid clouds*, where local resources of an organization that owns a private cloud is complemented with public cloud resources to meet spikes in resource demand;
- Multicloud*, when the end-user coordinates access and utilization of different cloud providers to meet his or her requirements; and
- Aggregated service by broker*, when a third-party (the broker) coordinates the access and utilization of multiple cloud resources on behalf of a user.

We also discussed standardization initiatives that are under development in the area of cloud interoperability and presented a comprehensive survey of research projects in this direction. As the summary Tables III and IV in online Appendix A indicate, even though there is a significant number of works in development, a few works address all the motivation scenarios and challenges we discussed. Therefore, it is possible that comprehensive and holistic approaches to cloud interoperability will be a result of the combination of one or more of the ongoing initiatives, either directly or via an extra abstraction layer hiding the complexities from end-users.

Moreover, from the summary of the projects, it is possible to notice that there are currently only a few cloud federation projects and they are mostly brokering technologies for multicloud and aggregated service scenarios. This is because so far, cloud services have been designed without considering cloud interoperability issues. We will see more of federated and hybrid cloud environments in the future, when more cloud providers will emerge with standard interfaces for their services.

Our summary also identified that, apart from scalability, avoidance of vendor lock-in is the most common motivation for Inter-cloud projects (Table III). This is because cloud computing users are vulnerable to rises in prices, decreases in availability, and even the cloud provider's bankruptcy and consequent loss of access to data stored on the provider. As a consequence, most current Inter-cloud projects are motivated by interoperability and avoidance of vendor lock-in. However, lock-in might be attractive to cloud providers as it enables them to retain their customers with little effort in having competitive products. Researchers and professionals who are working in the interconnected cloud area must take into account that, although avoidance of vendor lock-in is the great motivation for customers, it does not provide enough incentive for providers to boost up clouds' integration. This is why a large group of professionals believe that clouds' integration must be enabled on a separated layer detached from both vendors and providers.

After the analysis of ongoing projects, we analyzed the state of the art and the trends in the area of integrated clouds, where we identified that legal issues and meeting regulations are major concerns that are not well studied by the current projects. Therefore, appropriate application brokering that honors legal issues in terms of SLA is necessary.

Apart from interfaces, we also concluded that issues regarding economic aspects of cloud interoperability have received little attention, even though interoperation cannot be achieved without the resolution of these economic aspects. Once cloud vendors are convinced that adoption of cloud interoperability awards them financial and economical benefits, the goal of ubiquitously interconnected clouds is more likely to be achieved. This requires addressing issues regarding billing and accounting, novel methods of pricing suitable for interconnected cloud environments, and finally formation of Inter-cloud marketplaces.

To summarize, our broad and deep analysis of challenges and issues regarding cloud interoperability shows a limited trace of agreement and completeness among existing projects. The current study is intended to pave the way for further research and development activities by identifying weaknesses and deriving guidelines toward the holistic approach for interconnected clouds.

ELECTRONIC APPENDIX

The electronic appendix for this article can be accessed in the ACM Digital Library.

ACKNOWLEDGMENTS

The authors would like to thank Amir Vahid Dastjerdi, Mohsen Amini Salehi, Deepak Poola, and Nikolay Grozev for their constructive comments and suggestions on improving this survey. They also wish to acknowledge the comments and contributions provided by three anonymous reviewers that greatly strengthened the manuscript.

REFERENCES

- Jamal Abawajy. 2009. Determining service trustworthiness in intercloud computing environments. In *Proceedings of the 10th International Symposium on Pervasive Systems, Algorithms, and Networks (ISPAN'09)*. 784–788.
- Roberto Alfieri, Roberto Cecchini, Vincenzo Ciaschini, Luca dell'Agnello, Akos Frohner, Alberto Gianoli, Karoly Lorentey, and Fabio Spataro. 2004. VOMS, an authorization system for virtual organizations. In *Grid Computing*, Francisco Fernández Rivera, Marian Bubak, Andrés Gómez Tato, and Ramón Doallo (Eds.). Lecture Notes in Computer Science, Vol. 2970. Springer, Berlin, 33–40.
- Alba Amato, Loredana Liccardo, Massimiliano Rak, and Salvatore Venticinque. 2012. SLA negotiation and brokering for sky computing. In *Proceedings of the 2nd International Conference on Cloud Computing and Services Science (CLOSER'12)*. SciTePress, Porto, Portugal, 611–620.
- Sergio Andreozzi, Natascia De Bortoli, Sergio Fantinel, Antonia Ghiselli, Gian Luca Rubini, Gennaro Tortone, and Maria Cristina Vistoli. 2005. GridICE: A monitoring service for grid systems. *Future Generation Computer Systems* 21, 4 (2005), 559–571.
- Alain Andrieux, Karl Czajkowski, Asit Dan, Kate Keahey, Heiko Ludwig, Toshiyuki Nakata, Jim Pruyne, John Rofrano, Steve Tuecke, and Ming Xu. 2004. *Web Services Agreement Specification (WS-Agreement)*. Technical Report.
- Tomonori Aoyama and Hiroshi Sakai. 2011. Inter-cloud computing. *Business & Information Systems Engineering* 3, 3 (2011), 173–177.
- Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy H. Katz, Andrew Konwinski, Gunho Lee, David A. Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. 2010. A view of cloud computing. *Commun. ACM* 53, 4 (2010), 50–58.
- Arutyun I. Avetisyan, Roy Campbell, Indranil Gupta, Michael T. Heath, Steven Y. Ko, Gregory R. Ganger, Michael A. Kozuch, David O'Hallaron, Marcel Kunze, Thomas T. Kwan, Kevin Lai, Martha Lyons, Dejan S. Milojicic, Hing Yan Lee, Yeng Chai Soh, Ng Kwang Ming, Jing-Yuan Luke, and Han Namgoong. 2010. Open Cirrus: A global cloud computing testbed. *Computer* 43, 4 (Apr. 2010), 35–43.
- Dominic Battré, Frances M. T. Brazier, Kassidy P. Clark, Michael Oey, Alexander Papaspyrou, Oliver Wäldrich, Philipp Wieder, and Wolfgang Ziegler. 2010. A proposal for WS-agreement negotiation. In *Proceedings of the 11th IEEE/ACM International Conference on Grid Computing (GRID'10)*. 233–241.
- David Bermbach, Markus Klems, Stefan Tai, and Michael Menzel. 2011. MetaStorage: A federated cloud storage system to manage consistency-latency tradeoffs. In *Proceedings of IEEE International Conference on Cloud Computing (CLOUD'11)*. Washington, DC, 452–459.
- David Bernstein, Erik Ludvigson, Krishna Sankar, Steve Diamond, and Monique Morrow. 2009. Blueprint for the InterCloud—protocols and formats for cloud computing interoperability. In *Proceedings of the 4th International Conference on Internet and Web Applications and Services*. 328–336.
- David Bernstein and Deepak Vij. 2010a. Intercloud directory and exchange protocol detail using XMPP and RDF. In *Proceedings of the 6th World Congress on Services (SERVICES'10)*. Miami, FL, 431–438.
- David Bernstein and Deepak Vij. 2010b. Intercloud security considerations. In *Proceedings of the 2nd IEEE International Conference on Cloud Computing Technology and Science (CloudCom'10)*. Indianapolis, IN, 537–544.

- David Bernstein and Deepak Vij. 2010c. Simple storage replication protocol (SSRP) for intercloud. In *Proceedings of the 2nd International Conference on Emerging Network Intelligence (EMERGING'10)*. 30–37.
- David Bernstein, Deepak Vij, and Stephen Diamond. 2011. An intercloud cloud computing economy - technology, governance, and market blueprints. In *Proceedings of 2011 Annual SRII Global Conference (SRII)*. IEEE, San Jose, CA, 293–299.
- Luiz F. Bittencourt, Carlos R. Senna, and Edmundo R. M. Madeira. 2010. Enabling execution of service workflows in grid/cloud hybrid systems. In *IEEE/IFIP Network Operations and Management Symposium Workshops (NOMS Wksp)*. 343–349.
- Lianne Bodenstaff, Andreas Wombacher, Manfred Reichert, and Micheal C. Jaeger. 2008. Monitoring dependencies for SLAs: The MoDe4SLA approach. In *Proceedings of IEEE International Conference on Services Computing (SCC'08)*, Vol. 1. Miami, FL, 21–29.
- Janine Anthony Bowen. 2010. *Cloud computing: Principles and paradigms*. Vol. 87. Wiley, Chapter Legal Issues in Cloud Computing, 593–613.
- Ivan Breskovic, Michael Maurer, Vincent C. Emeakaroha, Ivona Brandic, and Jörn Altmann. 2011. Towards autonomic market management in cloud computing infrastructures. In *Proceedings of the International Conference on Cloud Computing and Services Science (CLOSER'11)*. 24–34.
- Rajkumar Buyya, Rajiv Ranjan, and Rodrigo N. Calheiros. 2010. InterCloud: Utility-oriented federation of cloud computing environments for scaling of application services. In *Proceedings of the 10th International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP'10)*, Vol. 6081. Busan, South Korea, 13–31.
- Rodrigo N. Calheiros, Adel Nadjaran Toosi, Christian Vecchiola, and Rajkumar Buyya. 2012a. A coordinator for scaling elastic applications across multiple clouds. *Future Generation Computer Systems* 28, 8 (2012), 1350–1362.
- Rodrigo N. Calheiros, Christian Vecchiola, Dileban Karunamoorthy, and Rajkumar Buyya. 2012b. The aneka platform and QoS-driven resource provisioning for elastic applications on hybrid clouds. *Future Generation Computer Systems* 28, 6 (2012), 861–870.
- Emanuele Carlini, Massimo Coppola, Patrizio Dazzi, Laura Ricci, and Giacomo Righetti. 2012. Cloud federations in contrail. In *Euro-Par 2011: Parallel Processing Workshops*. Lecture Notes in Computer Science, Vol. 7155. Springer, Berlin, 159–168.
- Antonio Celesti, Francesco Tusa, Massimo Villari, and Antonio Puliafito. 2010a. How to enhance cloud architectures to enable cross-federation. In *Proceedings of the 3rd International Conference on Cloud Computing (Cloud'10)*. Miami, FL, 337–345.
- Antonio Celesti, Francesco Tusa, Massimo Villari, and Antonio Puliafito. 2010b. Improving virtual machine migration in federated cloud environments. In *Proceedings of the 2nd International Conference on Evolving Internet (INTERNET'10)*. 61–67.
- Antonio Celesti, Francesco Tusa, Massimo Villari, and Antonio Puliafito. 2010c. Security and cloud computing: InterCloud identity management infrastructure. In *Proceedings of the 19th IEEE International Workshop on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE'10)*. 263–265.
- Antonio Celesti, Massimo Villari, and Antonio Puliafito. 2010d. A naming system applied to a RESERVOIR cloud. In *Proceedings of the 6th International Conference on Information Assurance and Security (IAS'10)*. Atlanta, GA, 247–252.
- Anirban Chakrabarti, Anish Damodaran, and Shubhashis Sengupta. 2008. Grid computing security: A taxonomy. *IEEE Security & Privacy* 6, 1 (2008), 44–51.
- David Chen and Guy Doumeingts. 2003. European initiatives to develop interoperability of enterprise applications - basic concepts, framework and roadmap. *Annual Reviews in Control* 27, 2 (2003), 153–162.
- Taesang Choi, Nodir Kodirov, Tae-Ho Lee, Doyeon Kim, and Jaegi Lee. 2011. Autonomic management framework for cloud-based virtual networks. In *Proceedings of the 13th Asia-Pacific Network Operations and Management Symposium (APNOMS'11)*. 1–7.
- Stuart Clayman, Alex Galis, Clovis Chapman, Giovanni Toffetti, Luis Rodero-Merino, Luis M. Vaquero, and Kenneth Naginand Benny Rochwerger. 2010. Monitoring service clouds in the future internet. In *Towards the Future Internet - Emerging Trends from European Research*. Amsterdam, Netherlands, 1–12.
- Antonio Cuomo, Giuseppe Modica, Salvatore Distefano, Antonio Puliafito, Massimiliano Rak, Orazio Tomarchio, Salvatore Venticinque, and Umberto Villano. 2013. An SLA-based broker for cloud infrastructures. *Journal of Grid Computing* 11, 1 (2013), 1–25.
- Amir Vahid Dastjerdi, Sayed Gholam Hassan Tabatabaei, and Rajkumar Buyya. 2012. A dependency-aware ontology-based approach for deploying service level agreement monitoring services in Cloud. *Software: Practice and Experience* 42, 4 (2012), 501–508.

- Scott Dowell, Albert Barreto, James Bret Michael, and Man-Tak Shing. 2011. Cloud to cloud interoperability. In *Proceedings of the 6th International Conference on System of Systems Engineering (SoSE'11)*. 258–263.
- Erik Elmroth and Lars Larsson. 2009. Interfaces for placement, migration, and monitoring of virtual machines in federated clouds. In *Proceedings of the 8th International Conference on Grid and Cooperative Computing (GCC'09)*. Lanzhou, China, 253–260.
- Erik Elmroth, Fermín Galán Marquez, Daniel Henriksson, and David P. Ferrera. 2009. Accounting and billing for federated cloud infrastructures. In *Proceedings of the 8th International Conference on Grid and Cooperative Computing (GCC'09)*. Lanzhou, China, 268–275.
- Ana Juan Ferrer, Francisco Hernández, Johan Tordsson, Erik Elmroth, Ahmed Ali-Eldin, Csilla Zsigri, Raúl Sirvent, Jordi Guitart, Rosa M. Badia, Karim Djemame, Wolfgang Ziegler, Theo Dimitrakos, Srijith K. Nair, George Kousiouris, Kleopatra Konstanteli, Theodora Varvarigou, Benoit Hudzia, Alexander Kipp, Stefan Wesner, Marcelo Corrales, Nikolaus Forgó, Tabassum Sharif, and Craig Sheridan. 2012. OPTIMIS: A holistic approach to cloud service provisioning. *Future Generation Computer Systems* 28, 1 (2012), 66–77.
- Brian W. Fitzpatrick and J. J. Lueck. 2010. The case against data lock-in. *Queue* 8, 10 (2010), 20:20–20:26.
- Íñigo Goiri, Jordi Guitart, and Jordi Torres. 2011. Economic model of a cloud provider operating in a federated Cloud. *Information Systems Frontiers* 14, 4 (2011), 827–843.
- Eduardo R. Gomes, Quoc Bao Vo, and Ryszard Kowalczyk. 2012. Pure exchange markets for resource sharing in federated Clouds. *Concurrency and Computation: Practice and Experience* 23, 9 (2012), 977–991.
- Ian Gorton, Yan Liu, and Jian Yin. 2010. Exploring architecture options for a federated, cloud-based system biology knowledgebase. In *IEEE 2nd International Conference on Cloud Computing Technology and Science (CloudCom'10)*. 218–225.
- Andrzej Goscinski and Michael Brock. 2010. Toward dynamic and attribute based publication, discovery and selection for cloud computing. *Future Generation Computer Systems* 26, 7 (2010), 947–970.
- Nikolay Grozev and Rajkumar Buyya. 2012. Inter-cloud architectures and application brokering: Taxonomy and survey. *Software: Practice and Experience* (2012), n/a–n/a. DOI: <http://dx.doi.org/10.1002/spe.2168>
- Seung-Min Han, Mohammad Mehedi Hassan, Chang-Woo Yoon, and Eui-Nam Huh. 2009. Efficient service recommendation system for cloud computing market. In *Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human (ICIS'09)*. 839–845.
- Piyush Harsh, Yvon Jegou, Roberto Casella, and Christine Morin. 2011. Contrail virtual execution platform challenges in being part of a cloud federation. In *Towards a Service-Based Internet*, Witold Abramowicz, Ignacio Llorente, Mike Surridge, Andrea Zisman, and Julien Vayssire (Eds.). Lecture Notes in Computer Science, Vol. 6994. Springer, Berlin, 50–61.
- Zach Hill and Marty Humphrey. 2010. CSAL: A cloud storage abstraction layer to enable portable cloud applications. In *Proceedings of 2nd IEEE International Conference on Cloud Computing Technology and Science (CloudCom'10)*. Indianapolis, IN, 504–511.
- Luokai Hu, Shi Ying, Xiangyang Jia Kai, and Zhao. 2009. Towards an approach of semantic access control for cloud computing. In *Cloud Computing*, Martin Gilje Jaatun, Gansen Zhao, and Chunming Rong (Eds.). Lecture Notes in Computer Science, Vol. 5931. Springer, Berlin, 145–156.
- Shigeru Imai, Thomas Chestna, and Carlos A. Varela. 2012. Elastic scalable cloud computing using application-level migration. In *Proceedings of the 5th IEEE/ACM International Conference on Utility and Cloud Computing (UCC'12)*. IEEE/ACM, 91–98.
- Katarzyna Keahey, Maurício Tsugawa, Andréa Matsunaga, and José A. B. Fortes. 2009. Sky computing. *IEEE Internet Computing* 13, 5 (2009), 43–51.
- Gabor Kecskemeti, Michael Maurer, Ivona Brandic, Attila Kertesz, Zsolt Nemeth, and Schahram Dustdar. 2012. Facilitating self-adaptable inter-cloud management. In *Proceedings of the 20th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP'12)*. Garching, Germany, 575–582.
- Attila Kertesz and Szilvia Varadi. 2014. Legal aspects of data protection in cloud federations. In *Security, Privacy and Trust in Cloud Systems*, Surya Nepal and Mukaddim Pathan (Eds.). Springer, Berlin, 433–455.
- Khaled M. Khan and Qutaibah Malluhi. 2010. Establishing trust in cloud computing. *IT Professional* 12, 5 (2010), 20–27.
- Hyunjoo Kim and Manish Parashar. 2011. CometCloud: An autonomic cloud engine. In *Cloud Computing: Principles and Paradigms*, Rajkumar Buyya, James Broberg, and Andrzej Goscinski (Eds.). Wiley, 275–298.
- Tobias Kurze, Markus Klemsy, David Bermbachy, Alexander Lenkz, Stefan Taiy, and Marcel Kunze. 2011. Cloud federation. In *Proceedings of the 2nd International Conference on Cloud Computing, GRIDs, and Virtualization*. 32–38.

- Kien Le, Ricardo Bianchini, Jingru Zhang, Yogesh Jaluria, Jiandong Meng, and Thu D. Nguyen. 2011. Reducing electricity cost through virtual machine placement in high performance computing clouds. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. Seattle, WA, 22:1–22:12.
- Maik Lindner, Fermin Galan, Clovis Chapman, Stuart Calyman, Daneil Henriksson, and Erik Elmroth. 2010. The cloud supply chain: A framework for information, monitoring, accounting and billing. In *Proceedings of the 2nd International ICST Conference on Cloud Computing (CloudComp'10)*. Springer Verlag, Barcelona, Spain.
- Marc X. Makkes, Canh Ngo, Yuri Demchenko, Rudolf Stijkers, Robert Meijer, and Cees de Laat. 2013. Defining intercloud federation framework for multi-provider cloud services integration. In *Proceedings of the 4th International Conference on Cloud Computing, GRIDs, and Virtualization*. Valencia, Spain, 185–190.
- Attila Csaba Marosi, Gabor Kecskemeti, Attila Kertesz, and Peter Kacsuk. 2011. FCM: An architecture for integrating IaaS cloud systems. In *Proceedings of the 2nd International Conference on Cloud Computing, GRIDs, and Virtualization*. IARIA, Rome, Italy, 7–12.
- Matthew L. Massie, Brent N. Chun, and David E. Culler. 2004. The ganglia distributed monitoring system: design, implementation, and experience. *Parallel Comput.* 30, 7 (2004), 817–840.
- Michael Menzel and Rajiv Ranjan. 2012. CloudGenius: Decision support for web server cloud migration. In *Proceedings of the 21st International Conference on World Wide Web (WWW'12)*. 979–988.
- Marian Mihailescu and Yong Teo. 2010a. A distributed market framework for large-scale resource sharing. In *Euro-Par 2010—Parallel Processing*, Pasqua D'Ambra, Mario Guarracino, and Domenico Talia (Eds.). Lecture Notes in Computer Science, Vol. 6271. Springer, Berlin, 418–430.
- Marian Mihailescu and Yong Teo. 2010b. Strategy-proof dynamic resource pricing of multiple resource types on federated clouds. In *Algorithms and Architectures for Parallel Processing*. Vol. 6081. Springer, Berlin, 337–350.
- Marian Mihailescu and Yong Meng Teo. 2010c. Dynamic resource pricing on federated clouds. In *Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid'10)*. Melbourne, Australia, 513–517.
- Marian Mihailescu and Yong Meng Teo. 2010d. On economic and computational-efficient resource pricing in large distributed systems. In *Proceedings of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid'10)*. 838–843.
- Rafael Moreno-Vozmediano, Rubén S. Montero, and Ignacio M. Llorente. 2012. IaaS cloud architecture: From virtualized datacenters to federated cloud infrastructures. *Computer* 45, 12 (2012), 65–72.
- Francesco Moscato, Rocco Aversa, Beniamino Di Martino, Teodor-Florin Fortis, and Victor Munteanu. 2011. An analysis of mOSAIC ontology for cloud resources annotation. In *Proceedings of the Federated Conference on Computer Science and Information Systems (FedCSIS'11)*. 973–980.
- Francesco Moscato, Rocco Aversa, Beniamino Di Martino, Massimiliano Rak, Salvatore Venticinque, and Dana Petcu. 2010. An ontology for the cloud in mOSAIC. In *Cloud Computing*. CRC Press, Chapter 20, 467–485.
- Kenneth Nagin, David Hadas, Zvi Dubitzky, Alex Glikson, Irit Loy, Benny Rochwerger, and Liran Schour. 2011. Inter-cloud mobility of virtual machines. In *Proceedings of the 4th Annual International Conference on Systems and Storage (SYSTOR'11)*. Haifa, Israel, 3:1–3:12.
- Mark Needelman. 2004. The shibboleth authentication/authorization system. *Serials Review* 30, 3 (2004), 252–253.
- David Núñez, Isaac Agudo, Prokopios Drogkaris, and Stefanos Gritzalis. 2011. Identity management challenges for intercloud applications. In *Secure and Trust Computing, Data Management, and Applications*, Changhoon Lee, Jean-Marc Seigneur, James J. Park, and Roland R. Wagner (Eds.). Communications in Computer and Information Science, Vol. 187. Springer, Berlin, 198–204.
- Daniel Nurmi, Richard Wolski, Chris Grzegorczyk, Graziano Obertelli, Sunil Soman, Lamia Youseff, and Dmitrii Zagorodnov. 2009. The eucalyptus open-source cloud-computing system. In *Proceedings of the 9th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid'09)*. 124–131.
- Giuseppe Papuzzo and Giandomenico Spezzano. 2011. Autonomic management of workflows on hybrid grid-cloud infrastructure. In *Proceedings of the 7th International Conference on Network and Services Management (CNSM'11)*. IFIP, Paris, France, 230–233.
- Manish Parashar, Moustafa AbdElBaky, Ivan Rodero, and Aditya Devarakonda. 2013. Cloud paradigms and practices for computational and data-enabled science and engineering. *Computing in Science & Engineering* 15, 4 (2013), 10–18.
- Przemyslaw Pawluk, Bradley Simmons, Michael Smit, Marin Litoiu, and Serge Mankovski. 2012. Introducing STRATOS: A cloud broker service. In *Proceedings of the 5th IEEE International Conference on Cloud Computing (CLOUD'12)*. 891–898.

- Juan M. Marín Pérez, Jorge Bernal Bernabé, Jose M. Alcaraz Calero, Felix J. García Clemente, Gregorio Martínez Pérez, and Antonio F. Gómez Skarmeta. 2011. Semantic-based authorization architecture for Grid. *Future Generation Computer Systems* 27, 1 (2011), 40–55.
- Dana Petcu. 2011. Portability and interoperability between clouds: Challenges and case study. In *Towards a Service-Based Internet*. Vol. 6994. Springer, Berlin, 62–74.
- Dana Petcu, Ciprian Craciun, Marian Neagul, Silviu Panica, Beniamino Di Martino, Salvatore Venticinque, Massimiliano Rak, and Rocco Aversa. 2011. Architecturing a sky computing platform. In *Towards a Service-Based Internet. ServiceWave 2010 Workshops*. Vol. 6569. Springer, Berlin, 1–13.
- Dana Petcu, Georgiana Macariu, Silviu Panica, and Ciprian Crăciun. 2013. Portable cloud applications—from theory to practice. *Future Generation Computer Systems* 29, 6 (2013), 1417–1430.
- Massimiliano Rak, Salvatore Venticinque, Tamas Mahr, Gorka Echevarria, and Gorka Esnal. 2011. Cloud application monitoring: The mOSAIC approach. In *Proceedings of the 3rd IEEE International Conference on Cloud Computing Technology and Science (CloudCom'11)*. 758–763.
- Rajiv Ranjan and Liang Zhao. 2011. Peer-to-peer service provisioning in cloud computing environments. *Journal of Supercomputing* 65, 1 (2011), 1–31.
- B. Rochwerger, D. Breitgand, E. Levy, A. Galis, K. Nagin, I. M. Llorente, R. Montero, Y. Wolfsthal, E. Elmroth, J. Caceres, M. Ben-Yehuda, W. Emmerich, and F. Galan. 2009. The Reservoir model and architecture for open federated Cloud computing. *IBM Journal of Research and Development* 53, 4 (2009), 1–11.
- Benny Rochwerger, Constantino Vázquez, David Breitgand, David Hadas, Massimo Villari, Philippe Massonet, Eliezer Levy, Alex Galis, Ignacio M. Llorente, Rubén S. Montero, Yaron Wolfsthal, Kenneth Nagin, Lars Larsson, and Fermín Galán. 2011. An architecture for federated cloud computing. In *Cloud Computing*. John Wiley & Sons, 391–411.
- Luis Rodero-Merino, Luis M. Vaquero, Victor Gil, Fermín Galán, Javier Fontán, Rubén S. Montero, and Ignacio M. Llorente. 2010. From infrastructure delivery to service management in clouds. *Future Generation Computer Systems* 26, 8 (2010), 1226–1240.
- Mohsen Amini Salehi, Bahman Javadi, and Rajkumar Buyya. 2012. QoS and preemption aware scheduling in federated and virtualized Grid computing environments. *Journal of Parallel and Distributed Computing* 72, 2 (2012), 231–245.
- Lutz Schubert, Keith Jeffery, and Burkhard Neidecker-Lutz. 2010. *The Future for Cloud Computing: Opportunities for European Cloud Computing Beyond 2010*. Technical Report.
- Peter Sewell, Paweł T. Wojciechowski, and Asis Unyapoth. 2010. Nomadic pict: Programming languages, communication infrastructure overlays, and semantics for mobile computation. *ACM Transactions on Programming Languages and Systems* 32, 4 (2010), 12:1–12:63.
- Mukesh Singhal, Santosh Chandrasekhar, Tingjian Ge, Ravi Sandhu, Ram Krishnan, Gail-Joon Ahn, and Elisa Bertino. 2013. Collaboration in multicloud computing environments: Framework and security issues. *Computer* 46, 2 (2013), 76–84.
- Borja Sotomayor, Rubén S. Montero, Ignacio M. Llorente, and Ian Foster. 2009. Virtual infrastructure management in private and hybrid clouds. *IEEE Internet Computing* 13, 5 (2009), 14–22.
- Ananth I. Sundararaj, Ashish Gupta, and Peter A. Dinda. 2004. Dynamic topology adaptation of virtual networks of virtual machines. In *Proceedings of the 7th Workshop on Languages, Compilers, and Runtime Support for Scalable Systems (LCR'04)*. ACM, 1–8.
- Hassan Takabi, James B. D. Joshi, and Gail-Joon Ahn. 2010. Security and privacy challenges in cloud computing environments. *Security Privacy* 8, 6 (2010), 24–31.
- Adel Nadjaran Toosi, Rodrigo N. Calheiros, Ruppa K. Thulasiram, and Rajkumar Buyya. 2011. Resource provisioning policies to increase IaaS provider's profit in a federated cloud environment. In *Proceedings of the 13th IEEE International Conference on High Performance Computing and Communications (HPCC'11)*. 279–287.
- Adel Nadjaran Toosi, Ruppa K. Thulasiram, and Rajkumar Buyya. 2012. Financial option market model for federated cloud environments. In *Proceedings of the 5th IEEE/ACM International Conference on Utility and Cloud Computing (UCC'12)*. IEEE/ACM, 3–12.
- Carlos A. Varela and Gul Agha. 2001. Programming dynamically reconfigurable open systems with SALSA. *SIGPLAN Notices* 36, 12 (2001), 20–34.
- Constantino Vázquez, Eduardo Huedo, Rubén S. Montero, and Ignacio Martín. Llorente. 2009. Dynamic provision of computing resources from grid infrastructures and cloud providers. In *Workshops at the Grid and Pervasive Computing Conference (GPC'09)*. IEEE, Geneva, Switzerland, 113–120.
- Salvatore Venticinque, Rocco Aversa, Beniamino Martino, Massimiliano Rak, and Dana Petcu. 2011. A cloud agency for SLA negotiation and management. In *Euro-Par 2010 Parallel Processing Workshops*, MarioR.

- Guarracino, Frédéric Vivien, Jesper Larsson Träff, Mario Cannatoro, Marco Danelutto, Anders Hast, Francesca Perla, Andreas Knüpfer, Beniamino Martino, and Michael Alexander (Eds.). Lecture Notes in Computer Science, Vol. 6586. Springer, Berlin, 587–594.
- David Villegas, Norman Bobroff, Ivan Rodero, Javier Delgado, Yanbin Liu, Aditya Devarakonda, Liana Fong, S. Masoud Sadjadi, and Manish Parashar. 2012. Cloud federation in a layered service model. *Journal of Computer and System Sciences* 78, 5 (2012), 1330–1344.
- Jens-Sönke Vöckler, Gideon Juve, Ewa Deelman, Mats Rynge, and Bruce Berriman. 2011. Experiences using cloud computing for a scientific workflow application. In *Proceedings of the 2nd International Workshop on Scientific Cloud Computing (ScienceCloud'11)*. ACM, 15–24.
- Matthias Winkler, Thomas Springer, and Alexander Schill. 2010. Automating composite SLA management tasks by exploiting service dependency information. In *Proceedings of IEEE 8th European Conference on Web Services (ECOWS'10)*. 59–66.
- Cheng-Zhong Xu, Jia Rao, and Xiangping Bu. 2012. URL: A unified reinforcement learning approach for autonomic cloud management. *Journal of Parallel and Distributed Computing* 72, 2 (2012), 95–105.
- Zehua Zhang and Xuejie Zhang. 2012. An economic model for the evaluation of the economic value of cloud computing federation. In *Future Communication, Computing, Control and Management*, Ying Zhang (Ed.). Lecture Notes in Electrical Engineering, Vol. 141. Springer, Berlin, 571–577.

Received March 2013; revised December 2013; accepted February 2014

The Promise & Challenges of Cloud Storage

The Promise of Cloud Storage

The Challenges of Cloud Storage

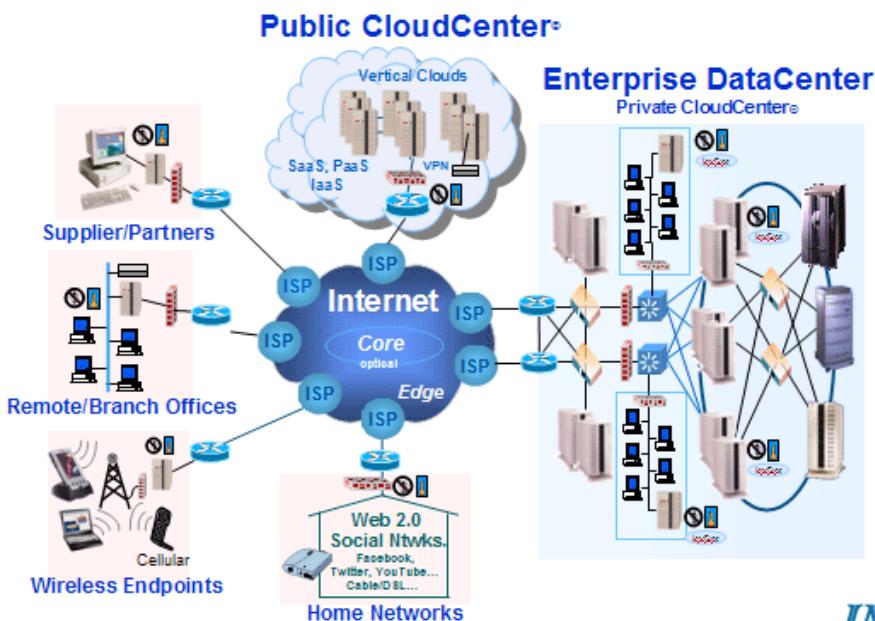
Types of Cloud Storage Solutions Available

Delivering an Information-Centric Cloud Storage Infrastructure

The Promise of Cloud Storage

With the onset of cloud computing, and more specifically cloud storage, organizations see potential in being able to reduce cost and complexity for these key applications. The promise of capacity on-demand – being able to pay only for what is used and for what is transferred – is appealing,

► Enterprise and Cloud Infrastructure (Secure, HA, Interoperable)



©2009-2010 IMEX Research All rights Reserved

IMEX
RESEARCH.COM

especially given that the price of (most) public cloud storage services are pennies on the dollar compared to traditional on-premises data center storage. The economies of scale and shared infrastructure allow service providers to deliver cloud-based storage at extremely low price points compared to that of a traditional infrastructure. But cloud-based computing and storage go far beyond this benefit by extending storage capabilities through the use of advanced API sets.

For IT customers, cloud-based storage is scalable, accessible, manageable and even more distributable, than a traditional storage infrastructure. Once the locality of data becomes irrelevant, users can integrate data from anywhere. And when the location of data is no longer important, it is easy to scale performance by distributing or moving data across any system according to demand. Exponential data growth particularly coming from unstructured data, is forcing companies to look at new, cost-effective ways to move their information from expensive primary disks to archive storage. Additionally an explosion of interest in large-scale, elastic and cloud-based data storage and processing using open-source Hadoop extract new value from both complex and structured data is being implemented both in-house and using cloud-based infrastructure.

Now cloud storage is being adopted as a reliable platform for long term archive needs given that new efficient tools are now available from vendors allowing users to quickly and securely move corporate information to the cloud.

Industry Report

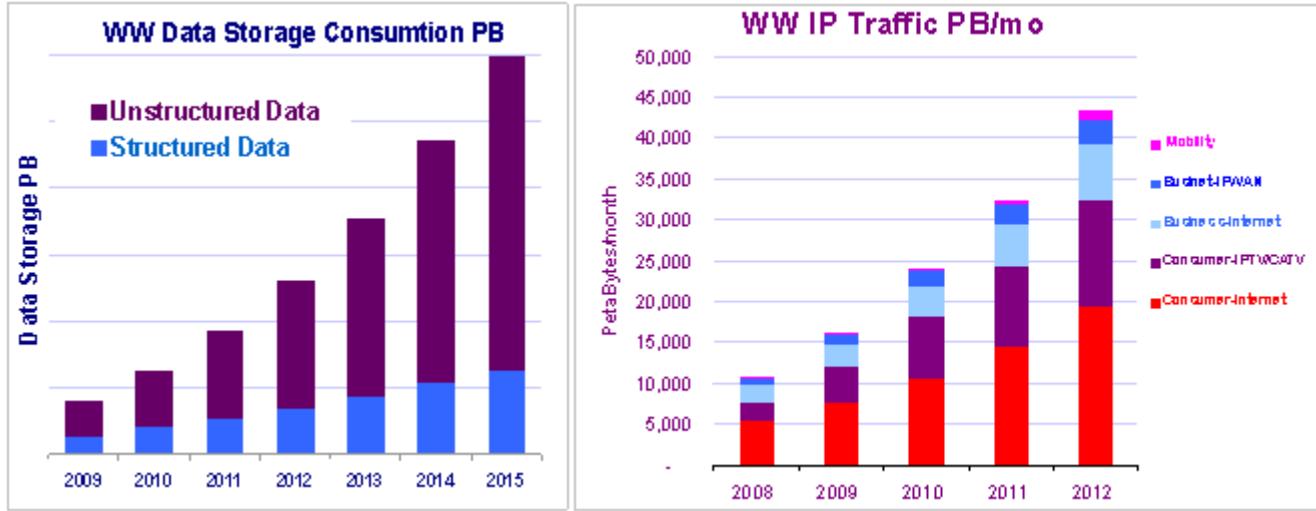
Table of Contents

- 1 Executive Summary**
- 2 Market Drivers/ Industry Dynamics**
- 3 Market Segmentation & Product Requirements**
- 4 Market Demand Forecast & Market Shares**
- 5 Enabling Technologies - Trends & Standards**
- 6 Competitive Product Positioning & Strategies**
- 7 Suppliers: Portfolios & Strategies**
- 8 Channels of Distribution**
- 9 Recommendations**
- 10 Methodology & Appendices**

IMEX
RESEARCH.COM

1474 Camino Robles Ct.
San Jose, CA - 95120
Tel: 408.268.0800
Fax: 408.268.2300

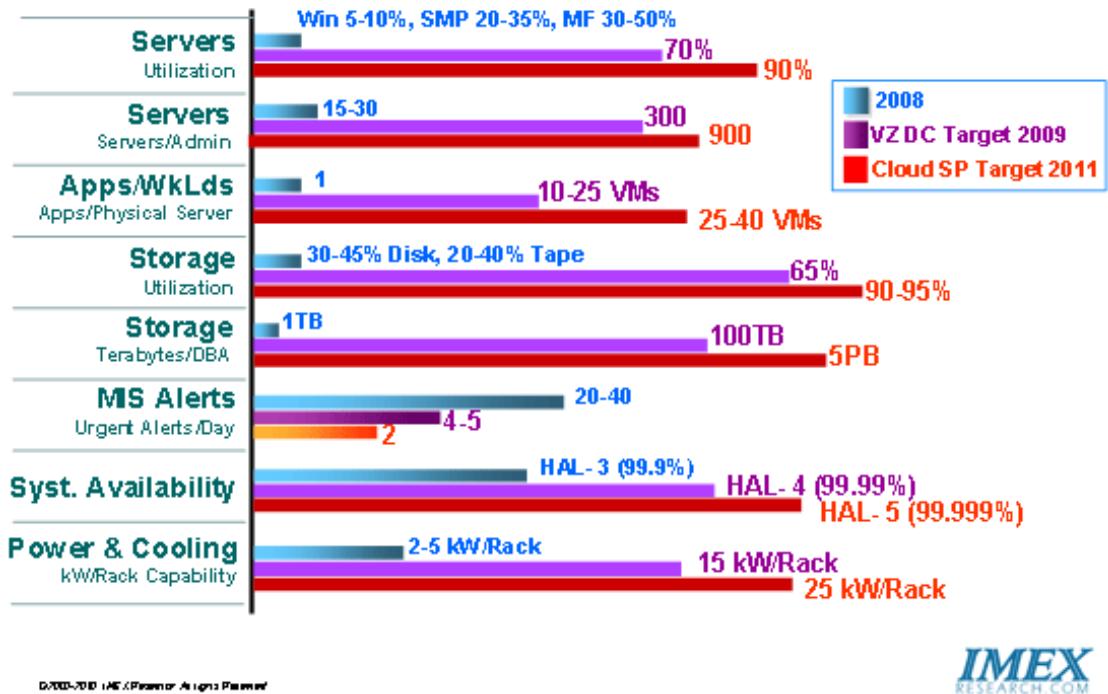
Click for email address
www.imexresearch.com



A typical example of emerging VertiClouds© reveals tremendous benefits of using cloud storage services. The U.S. healthcare system spends \$10–15 billion each year on redundant radiological exams that stem from physicians' lack of access to patients' prior imaging exams. Medical Imaging brings with it a significant and never ending demand for storage. The cloud computing and storage infrastructures and

Vertical Cloud© platforms create an opportunity to address the significant need for providing a consistent set of networked healthcare services for radiology, cardiology and patient records transport and management to enable sharing diagnostic imaging information. By using a 10GbE network from end-users to Cloud, ultimately supports thousands of hospitals, imaging centers and hundreds of thousands of physicians and millions of patients. In turn, users benefit from a virtualized, secure, reliable and elastic nature of cloud services ensures a capability to scale the performance and operation of the network for peak usage periods. Also the pay-for-use cloud storage model provides an affordable model for health care facilities of different sizes.

New Metrics for Cloud-Aware VZ DC Infrastructure



IMEX
RESEARCH.COM

Cloud Storage changes the game for distributed enterprises by delivering LAN-like application performance over the WAN and enabling a wave of IT consolidation from the branch into the data center or private cloud through optimizing network access for cloud storage deployments with new solutions. Now, enterprises are beginning to migrate storage to the public cloud to take advantage of the dramatic operational and cost benefits.

Having implemented virtualization in their data centers, organizations are now establishing creating new metrics and goals metrics to make their data centers cloud-ready.

The Challenges of Cloud Storage

Given the obvious benefits of Cloud Storage, raises the question as to why haven't more organizations already adopted cloud storage for on-premises applications and alleviate present expensive storage.

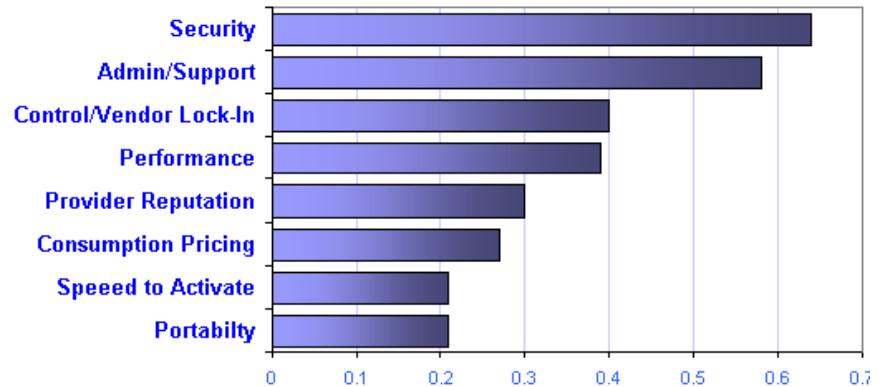
The answer – Right now, there are traditional IT barriers which must be overcome when considering integration of cloud storage into today's production environments.

- Performance and data transfer rates become key issues as the distance between the data and the user increases - which is what happens in cloud computing.
- Even unlimited bandwidth without solving the latency problem will not improve the performance because it is the latency - or the chattiness - of the protocols, plus the speed of light limitations that cause the end user experience to be very poor.
- Not all data access patterns are well suited to the cloud, particularly if there are large distances to cover. In such cases, bandwidth becomes not only a challenge but a financial consideration.
- Cloud storage isn't about to replace the storage network in the data center any time soon, at least not for data-intensive, high-performance, low-response time, transactional applications and for mission-critical data. But we will see many use cases where companies and organizations of all sizes will augment their on-premise storage with cloud storage potentially from various vendors in a hybrid model deployment. However, hybrid models tend to bring interoperability issues and the need to deal with different tools, API's, management infrastructure etc.
- The popular storage use cases tend to be infrequently accessed data scenarios including archiving, backup, DR, and offsite data protection. Performance and data transfer rates become key issues as the distance between the data and the user increases - which is what happens in cloud computing.
- It will be a hybrid cloud storage world for many years to come with a lot of that storage living in private clouds. Many large enterprises deal with petabytes of data and their processing.

A survey conducted by IMEX Research reveals that both enterprises and SMBs today are concerned about:

- **Security & Data Availability** - Concerned about the security of their data, once it is in the hands of the cloud provider whose multi-tenancy infrastructure is shared by others. Numerous questions arise when considering cloud storage, including "is my data secure?", "am I in compliance?", "what happens if a provider loses a disk drive?", "is my data protected?", "how do I know if my data is truly unusable when I delete it?", and many others. StorSimple allows you to provide an encryption key – or we can generate one for you – and all data sent to the cloud is encrypted. They are rightfully worried about the availability of their data and how that impacts their day-to-day operations. What happens if a cloud storage service is offline for a period of time?
- **Performance** - Legitimately concerns about application performance if the application storage is in the cloud. Will the cloud storage service satisfy my workloads?
- **Bandwidth limitations** – Bandwidth is a limiting factor when accessing a public storage cloud, as they are accessed over the Internet. Primary storage deduplication and compression, minimizes bandwidth consumption dramatically while also improving performance.
- **Latency constraints** – Latency is the silent killer of application performance, both in terms of response time and throughput. StorSimple takes advantage of parallelization, persistent connections, and TCP optimizations to overcome latency and improve performance
- **Manageability** - Are concerned about being locked into their proprietary cloud storage infrastructure and applications services. They don't have vendor independent tools or industry standards to evaluate the applicability or measure the effectiveness of cloud storage for their environment.
- **Interoperability/Protocol translation** – A serious concern exists today is: Most of today's on-premises applications use block protocols such as FC, iSCSI etc. But Cloud storage protocols predominantly speak only in the language of file protocols (CIFS, NFS) and both public and private storage clouds are accessed via REST HTTP-based, or SOAP APIs. Since these applications expect block access to storage, introducing a cloud storage system to the application is like trying to have a conversation in Spanish when you only speak English. So how will current applications even be interoperable between existing storage infrastructure and cloud storage? Taking advantage of these conversion tools and seamlessly integrating with cloud storage eliminates the need for complex application re-programming and integration (see below - Importance of Cloud Storage API set).
- **Costs** - Many cloud storage services charge their customers based on the amount of storage capacity consumed and the number of IOPs performed or the amount of bandwidth consumed. Much as reducing cloud storage costs through deduplication and optimization and taking advantage of pay-as-you-grow schemes helps but choosing a cloud storage vendor which meets all other criteria remains a challenge.

Cloud Computing Adoption Obstacles



Delivering an Information-Centric Cloud Storage Infrastructure

Organizations that presently use on-premise storage and wish to selectively federate data to public clouds, would need to create an integration that enables automated searching and tagging of on-premise data and creates a policy-based federation of data to public clouds to deliver a truly information-centric cloud infrastructure keeping in mind the following:

Organization Goals for Primary Storage

In area of cloud storage, organization goals for primary storage include:

- **Improved storage economics** – Leverage appropriate tiers of storage according to the performance requirement of the data.
- **Performance consistency** – working set and hotspot data to be automatically tiered to the highest-performance tier of storage (SSD), whereas non working-set or non hotspot data is tiered to a lower performing tier dynamically
- **Transparent integration** – no complex policies to manage, no agents to install on servers, and the storage view from server to remain same in light of new integration with cloud storage
- **Deduplication** - of primary storage to eliminate the repeated storage of redundant segments of data. Deduplication to be completely transparent to the server, i.e. the server's view of the content of their storage volumes should completely remain unchanged.
- **Compression** - Achieve higher levels of compression even when encountering single-byte insertion scenarios.
- **Data Reduction** - Achieve 10X data reduction for specific storage workloads.
- **Databases** - Separate the storage of BLOBs (Binary Large OBjects) from content databases in conjunction with appropriate application frameworks . BLOB externalization leads to smaller database sizes, reduced fragmentation and maintenance cycles, and consistent response times as the deployment scales
- **Performance consistency** –Hotspot elimination – Use technologies to automatically identify working sets, create integrated storage through deploying SSDs for hotspots removal and improve performance while leveraging , and low cost SATA for compressed, deduplicated data.
- **Version control support** – Version control can be enabled, thanks to some companies' implementation of primary storage deduplication, which minimizes storage capacity requirements and cost
- **Pay-as-you-grow** – use of cloud storage enables pay-as-you-grow capacity consumption, which minimizes cost of over-provisioned yet unutilized on-premises storage
- **Optimal Cost Structure** – Using deduplication, compression and encryption minimize cost for cloud storage capacity and cost for IO and data transfer,
- **Tape elimination** – Use Cloud Clones to enable consistent, point-in-time recovery snapshots and store independent copies in the cloud to eliminate need for tape
- **Advance API-enabled Capabilities** – Leverage advanced API sets to achieve following capabilities: Content search capabilities, Optimized Retrieval, Retention and Compliance Controls, Metadata Tagging, Security and others.

Application Performance Optimization using Data Segmentation

One of the key requirements to improve performance of data access and computations is to identify application/workload characteristics (IOPs, latency and bandwidth), data usage patterns (including frequency of access, I/Os, age, priority and relationship with other data) as well as data management (data placements, storage device characteristics and locality of reference) and then store the data on appropriate tier of storage and eliminate access hotspots.

Cloud Storage - Ideal for Certain Data Types

Cloud Storage - a poor fit in following data types:

Transactional Data

- Frequent Read and Write Accessed Data (eg DB)
- Massive I/O requirements
- Database, Souce Code, Active VMware images

Active Corporate Data

- Advanced Data Protection Schemes
- Office Docs and spreadsheets
- Source Code

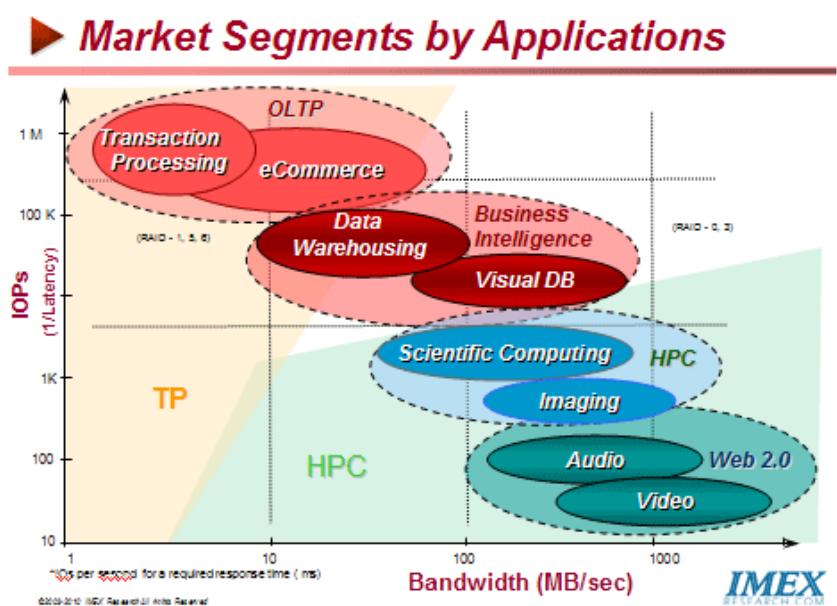
Cloud Storage - good for the following data types:

Large files with mostly read accesses

- Digital Content, Streaming Media, Video, Music
- Parallel Streaming Writes
- Video Surveillance (Private Clouds)

Long-Term Storage Files

- Backup and archival Files (Private Clouds)
- Medical Images, Energy Exploration, Genomics
- VMWare Backups



Geographically Shared Files

- Universal Access from anywhere, anytime (Public Clouds)
- Movie Trailers, Training Videos
- Corporate Marketing and Training Colaterals (docs/podcasts/videos)

Achieving Optimized Cloud Storage using Tiered Data Sets

Key to optimizing I/O performance of storage is understanding the characteristics of Data set volumes and taking advantage of tiered storage on-premise plus cloud storage services to achieve CAPSIMS© goals of

- **Cost** (Achieve 10X data reduction for specific storage workloads. Deduplication of primary storage to eliminate the repeated storage of redundant segments of data, Achieve higher levels of compression, Enabling Version control to minimize storage capacity requirements and cost, use of cloud storage enables pay-as-you-grow capacity consumption, which minimizes cost of over-provisioned yet unutilized on-premises storage.)

- **Availability** (Cloud storage D2D data replication cloning to enable consistent, point-in-time recovery snapshots and store independent copies in the cloud to eliminate need for tape)

- **Performance** (Leverage appropriate tiers of storage according to the performance requirement of the data, working set and hotspot data to be automatically tiered to the highest-performance tier of storage (SSD), whereas non working-set or non hotspot data is tiered to a lower performing tier dynamically (SATA) Separate the storage of Binary Large Objects from content databases in conjunction with appropriate application frameworks. BLOB externalization leads to smaller database sizes, reduced fragmentation and maintenance cycles, and consistent response times as the deployment scales)

- **Security** (Leverage advanced API sets to achieve Security)

- **Interoperability** (Leverage advanced API sets to achieve following capabilities: Content search capabilities, Optimized Retrieval, Retention and Compliance Controls, Metadata Tagging, Security and others)

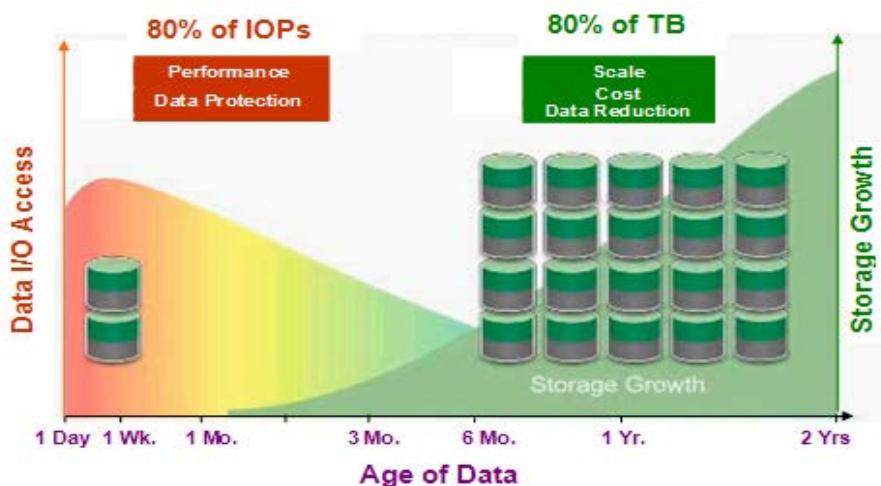
- **Manageability** (no complex policies to manage, no agents to install on servers, and the storage view from server to remain same in light of new integration with cloud storage, Leverage advanced API sets to achieve following capabilities: Content search capabilities, Optimized Retrieval, Retention and Compliance Controls, Metadata Tagging, Security and others.)

- **Solid State Drives (SSD)** – SSD provides the highest levels of throughput and the lowest response times. WSL helps ensure that the working set for a given volume resides on SSD to ensure consistent, high performance access

- **FC/FCoE/iSCSI/SAS** – Fibre Channel SANs are predominantly used in large enterprises while SMBs are more inclined towards using iSCSI SANs for primary storage and to achieve higher availability and performance from their storage systems.

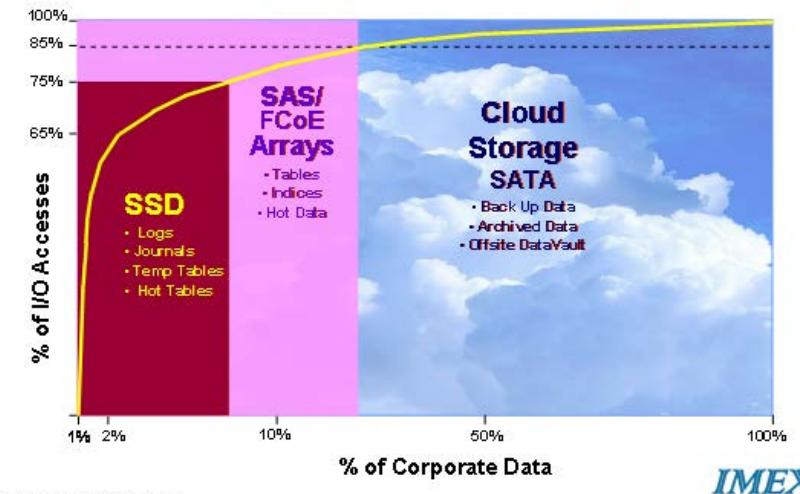
- **Serial ATA (SATA)** – SATA provides the secondary tier of internal storage, and is used for holding deduplicated working set and non working-set data. SATA provides high capacity and consistent levels of performance and response time

- **Cloud Storage** – Public or private cloud storage can be configured for a particular volume if so desired, and when enabled, is used as the lowest tier of storage for data. Cloud storage enables pay-as-you-grow capacity and the greatest levels of economic efficiency



Corporate Cloud Storage Usage

I/O Access Frequency vs. Percent of Corporate Data



IMEX
RESEARCH.COM

Types of Cloud Storage Solutions

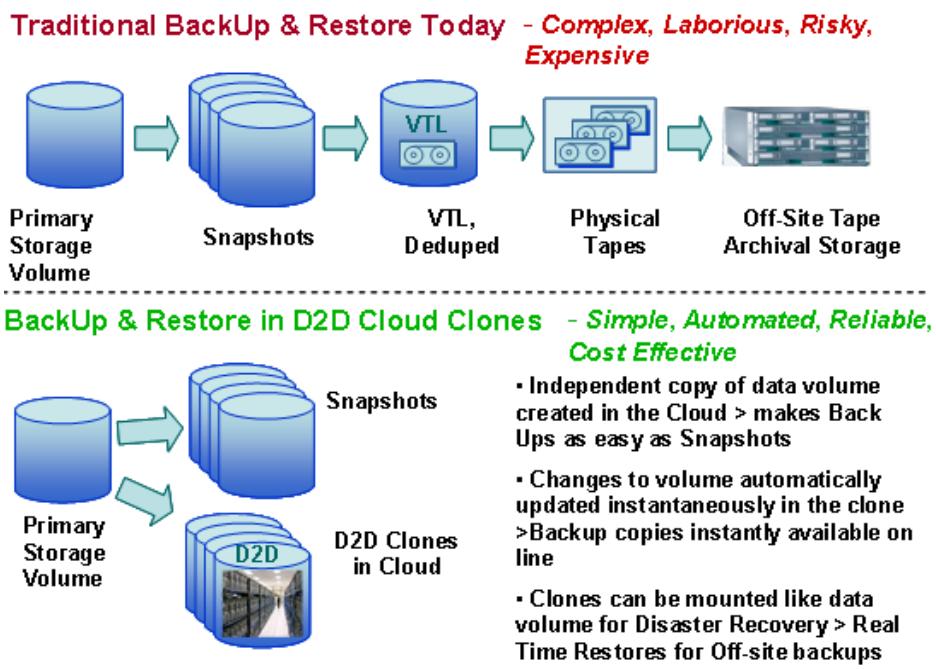
Customers need to decide whether or not a cloud storage service, public or private, should be used for a given data set and which cloud storage service class should be used before deciding to migrate to cloud storage services.

- **Classic DataCenter Storage** – Data set volumes configured as on-premise only do not take advantage of a cloud storage service but rather only take advantage of tiered storage system.
- **Private Cloud Storage** – Data set volumes configured in this mode take advantage of tiered storage on-premise plus a private cloud storage service on-premise that is used for data protection using a D2D Cloud Clone feature format compatible with public clouds. This helps minimize data protection infrastructure and cost
- **Federated Cloud Storage** – Data set volumes configured in this mode take advantage of both on-premises tiered storage and capacity made available through a cloud storage provider that is used for data protection

Cloud Clones are similar to snapshots in that they are a point-in-time, consistent copy of a series of application-related volumes. Whereas snapshots are typically mounted by a backup server and written to virtual or physical tape, which is then shipped off-site for protection, Cloud Clones on the other hand are stored persistently in the cloud. They are a replacement for VTL and tape libraries and allow one to mount and restore data at any point in time without having the hassle of retrieving tapes. Cloud Clones are de-duplicated and compressed, minimizing cloud capacity and data transfer costs, which makes them more cost compelling than tape as a form of archival storage. Cloud Clones are encrypted using a key the user supplies (or optionally the system can generate for the user) to ensure confidentiality while data sets are stored in the cloud using cloud clones.

Features of a good Cloud Storage Solution

- Fully integrates into traditional on-site storage as well as the cloud so users who want to store large amounts of content can connect to either in minutes



- A hybrid storage solution that makes cloud storage appear like local data center storage that integrates into customers' existing storage and data management tools. It identifies and stores all the hotspot and bottleneck data on a tier of high-performance Solid State Drives (SSD), enabling the use of lower-cost SATA storage and/or cloud storage as primary storage. It also performs real-time data deduplication to minimize the footprint of the data stored and provides the WAN optimization functions for cloud storage. All data that is stored in the cloud is encrypted.
- It intelligently identifies the bottleneck areas and stores the data on a tier of integrated high-performance Solid State Drives (SSD), thereby, allowing them to use low cost SATA storage and/or cloud storage as primary storage for the data needs. Their technology transparently examines all data, breaking it into smaller segments, and stores the data segments on the appropriate tier of storage according to its frequency of use, age, relationship with other segments, and

priority.

- To transparently protect and manage file, email and other applications' data copies across on premises and public cloud storage to improve service levels, while simplifying and greatly reducing data management costs.
- Native integration with the REST based API also delivers increased performance over alternate protocols. It also enables access to Cloud Storage from native CIFS or NFS clients.
- Turns Windows Explorer into a Cloud Storage Portal using REST API on Cloud Desktops, enabling native access and providing an ability to access content as if it were stored on the desktop with drag and drop migration.
- SMBs can now enhance their file servers to store content in the Cloud while implementing a smart cache that maintains LAN access speeds.
- Identifies and archives inactive data from applications and data warehouses—including master, reference and transactional data—to Cloud Storage Service that is readily accessible when needed, streamlining information lifecycle management while reducing management, software, and hardware costs.
- A simplified virtual file server that delivers cost-effective and completely secure cloud storage offsite for businesses yet retaining the local performance and functionality of a traditional NAS

- Provide a specialized data retention repository in which to preserve large amounts of historical data for regulatory or business purposes
- Key is to create an archive file system with the unified goals of providing unfettered access to Cloud Storage as an enterprise archiving storage tier on an Exabyte level scale by presenting the cloud as a standard network share while removing the hurdles associated with REST or SOAP integration.

Hybrid Storage – A new class of Storage Solution

Now a new class of storage solutions – namely hybrid storage solutions – are emerging that help address these exact challenges. A hybrid storage solution is:

- Deployed in the customer's data center
- Provides servers with access to storage using protocols that they understand
- Speaks cloud storage service protocols, and virtualizes cloud storage into usable data center capacity
- Overcomes performance limitations associated with cloud storage
- Addresses security concerns associated with cloud storage

These solutions typically have the following characteristics:

- They are application-aware and tune the storage system's behavior according to the application's needs.
- They have multi-class integrated storage with complicated caching algorithms or simple automated data tiering.
- They tend to operate like any one of the storage arrays that already exist in the data center.
- They sit between the servers and the cloud, and allow users to control where the data is stored.

Reduced Storage Costs

These solutions help address the immediate requirement for adding capacity inexpensively. Aside from the cost of the solution hardware, the rest of expense on storage is based on how much capacity the user utilizes in the cloud and how much data they transfer (These rates vary from cloud services providers such as Amazon Web Services, Rackspace etc.)

Application Aware Tiering

They have application-awareness that helps to address specific storage-related issues that impact the ability of an organization to scale or manage an application. This can be manifest in application-aware tiering, volume data location policies, and even plug-ins that integrate with frameworks provided by the application vendor.

Storage Efficiency

They provide primary storage deduplication, which not only helps improve performance when using the cloud, but also minimize the cloud cost (since only deduplicated data would ever be read from or written to the cloud).

They also help address some of the concerns with Exchange and SharePoint regarding storage efficiency which is especially important, as Microsoft removed the Single Instance Store for attachments in Exchange 2010 and also for using versioning and extended recycle bins in SharePoint. Deduplication ensures that redundant email attachments are stored in a more space-efficient manner, meaning lower storage capacity consumption

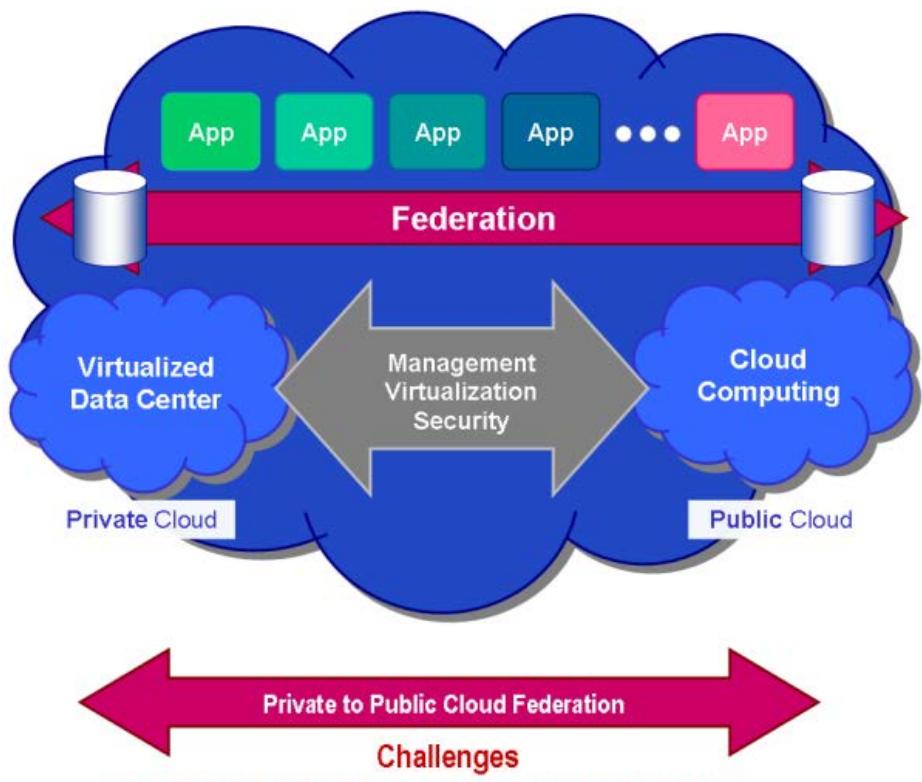
Security

They allow users to control the encryption key, and it is never shared with the cloud provider. This increases confidence, as organizations don't need to worry about what happens if someone gets a hold of their data from the cloud, or if the cloud provider is requested to release the data – or hardware where data may reside – in support of investigation or litigation.

Coupled with other security services provided by the cloud provider, such as Virtual Private Networks, roles-based access control, multi-factor authentication etc., cloud storage can be as secure as traditional on-premises storage for certain applications. For these extra security features additional dedicated performance-boosting power is generally needed.

Data Protection

Helping simplify data protection is one of the major benefits of Hybrid Storage Solutions. Having a virtually unlimited pool of storage sitting behind an on-premises hybrid storage appliance that provides data deduplication and encryption means that with the right application hooks and validation with data protection software vendors, users can take snapshots - that are not only crash-consistent



Interoperability, Data Availability (Security, Uptime), Consistency, Performance (Latency, Bandwidth), Management (Advanced APIs)

but also application-consistent - and store them in a space-efficient, secure format on the cloud as an always-on backup and available in case needed for disaster-recovery. So instead of having to continually fetch tape from an off-site vault, users can simply use a cloud backup copy to perform a mailbox restore, object restore or even recover the entire application.

Seamless Integration of Data Storage between Data Centers & Cloud

Usually, when organizations decide to leverage cloud based storage solutions, they will be forced to rewrite their applications because most of the cloud storage providers give access to their storage using http or some REST based APIs. Traditional applications are not capable of accessing the storage that uses REST based API and it will require a rewrite of the applications or deployment of some sort of a connector. A start up company mitigate this using hybrid storage solutions that makes cloud storage from multiple vendors such as EMC, IronMountain, Microsoft, and Amazon, appear like local data center storage that integrates into customers' existing storage and data management tools.

Much as these solutions are relatively nascent, many customers are achieving lower total cost of ownership, simplified backup and restore, better disaster recovery, and the ability to confidently scale many applications without concern over performance.

Advanced Cloud Storage API enabled Capabilities

A cloud NAS gateway that many storage providers offer, provides an important translation between data center network storage protocols (e.g. CIFS/NFS) and a more internet-optimized protocol (e.g. WebDAV) that is less chatty. The cloud NAS gateway provides a viable quick fix to get data moving to the cloud. The advantage of a cloud NAS is that it enables a customer to begin archiving data to the cloud almost as easily as they could send data to a local network mounted disk. This allows for rapid adoption of cloud storage as an archive destination. But interacting and performing efficient communication to the cloud is just the bare minimum of what an API should do. Advanced cloud storage API sets allow following improved capabilities:

Content Search Capability – An API set from within the programming language can provide the ability to deliver context searching during ingestion on the data that is being stored on it and create the index - all done on the back end, by the cloud storage system. Subsequent commands allow the application to query the index and retrieve results. These queries can be against the core content itself, or they can be metadata specified during ingestion and even display the results. This avoids having to issue subsequent retrieval requests to paint the search results. The query API set can return a set or a "page" at a time results, allowing the user browse through very large search results sets.

Optimized Retrieval - Advanced API sets can also provide commands to the cloud storage to store the data in a parsed format allowing an organization to retrieve just the components of the data that is needed. Having these items individually addressable provides an important workaround to mitigate the latency of the cloud. A typical example would be to command the API to parse the metadata (header, body and attachments) only. On search only the relevant metadata (header and or body) is pulled across the internet. With relevant email's header found, associated large attachments can then be brought across the internet.

Retention and Compliance Controls - The API set should be enable to set retention and WORM levels within the application itself optionally set by an administrator or user or automatically be activated tied to an internal analysis of the data. Retention commands should include the ability to set length of time to be stored, number of copies to be kept and if the data can be modified and allow for the assured destruction of data, confirming that the data is no longer recoverable.

Metadata Tagging - Metadata tagging by an API set allows for keywords or reference points to be set on files to help with search criteria, retention criteria or compliance criteria. Being able to build search optimization, pre-parsing data for retrieval, setting retention, compliance and metadata upfront at the point of archive goes a long way toward those feature sets actually being used.

A feature-rich API set allows building classification into the process, enabling an interactive archive to become an upfront, quick task performed consistently will likely increase the accuracy of the classification as the application usually has the most context at that point compared to traditionally classification projects that are very large and involve processing of data long after it was written with specifics of that data forgotten.

Security – The API data should be encrypted during transmission and remain encrypted when stored at the cloud storage provider's facility. Not only does this prevent the infiltration of data in flight, it also insures that only that customer can read their data. The API set should have the ability to store multiple copies of that data not only locally but in geographically disperse locations as well, providing a comfort level in recovery from a failure or data corruption locally as well as globally.

Click on the following for additional information or go to <http://www.imexresearch.com>



IMEX Research, 1474 Camino Robles San Jose, CA 95120 (408) 268-0800 <http://www.imexresearch.com>
If you wish not to receive any IT information/news from IMEX, please [click here](#).



**National Institute of
Standards and Technology**
U.S. Department of Commerce

Special Publication 800-145

The NIST Definition of Cloud Computing

Recommendations of the National Institute of Standards and Technology

Peter Mell
Timothy Grance

NIST Special Publication 800-145

The NIST Definition of Cloud Computing

Peter Mell

Timothy Grance

C O M P U T E R S E C U R I T Y

Computer Security Division
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899-8930

September 2011



U.S. Department of Commerce

Rebecca M. Blank, Acting Secretary

National Institute of Standards and Technology

Patrick D. Gallagher, Under Secretary for Standards and
Technology and Director

Reports on Computer Systems Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analysis to advance the development and productive use of information technology. ITL's responsibilities include the development of technical, physical, administrative, and management standards and guidelines for the cost-effective security and privacy of sensitive unclassified information in Federal computer systems. This Special Publication 800-series reports on ITL's research, guidance, and outreach efforts in computer security and its collaborative activities with industry, government, and academic organizations.

National Institute of Standards and Technology Special Publication 800-145
7 pages (September 2011)

Certain commercial entities, equipment, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

Acknowledgements

The authors Peter Mell and Timothy Grance of the National Institute of Standards and Technology (NIST) would like to thank the many experts in industry and government who contributed their thoughts to the creation and review of this definition. We especially acknowledge Murugiah Souppaya and Lee Badger, also of NIST, and Wayne Jansen of Booz Allen Hamilton, whose advice and technical insight assisted this effort.

Errata

The following changes have been incorporated into Special Publication 800-145, as of the date indicated in the table.

DATE	TYPE	CHANGE	PAGE NUMBER
4/27/2012	Editorial	Corrected page number from "2" to "1"	1

1. Introduction

1.1 Authority

The National Institute of Standards and Technology (NIST) developed this document in furtherance of its statutory responsibilities under the Federal Information Security Management Act (FISMA) of 2002, Public Law 107-347.

NIST is responsible for developing standards and guidelines, including minimum requirements, for providing adequate information security for all agency operations and assets; but such standards and guidelines shall not apply to national security systems. This guideline is consistent with the requirements of the Office of Management and Budget (OMB) Circular A-130, Section 8b(3), "Securing Agency Information Systems," as analyzed in A-130, Appendix IV: Analysis of Key Sections. Supplemental information is provided in A-130, Appendix III.

This guideline has been prepared for use by Federal agencies. It may be used by nongovernmental organizations on a voluntary basis and is not subject to copyright, though attribution is desired.

Nothing in this document should be taken to contradict standards and guidelines made mandatory and binding on Federal agencies by the Secretary of Commerce under statutory authority, nor should these guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the OMB, or any other Federal official.

1.2 Purpose and Scope

Cloud computing is an evolving paradigm. The NIST definition characterizes important aspects of cloud computing and is intended to serve as a means for broad comparisons of cloud services and deployment strategies, and to provide a baseline for discussion from what is cloud computing to how to best use cloud computing. The service and deployment models defined form a simple taxonomy that is not intended to prescribe or constrain any particular method of deployment, service delivery, or business operation.

1.3 Audience

The intended audience of this document is system planners, program managers, technologists, and others adopting cloud computing as consumers or providers of cloud services.

2. The NIST Definition of Cloud Computing

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. This cloud model is composed of five essential characteristics, three service models, and four deployment models.

Essential Characteristics:

On-demand self-service. A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider.

Broad network access. Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, tablets, laptops, and workstations).

Resource pooling. The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or datacenter). Examples of resources include storage, processing, memory, and network bandwidth.

Rapid elasticity. Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be appropriated in any quantity at any time.

Measured service. Cloud systems automatically control and optimize resource use by leveraging a metering capability¹ at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service.

Service Models:

Software as a Service (SaaS). The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure². The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.

Platform as a Service (PaaS). The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming

¹ Typically this is done on a pay-per-use or charge-per-use basis.

² A cloud infrastructure is the collection of hardware and software that enables the five essential characteristics of cloud computing. The cloud infrastructure can be viewed as containing both a physical layer and an abstraction layer. The physical layer consists of the hardware resources that are necessary to support the cloud services being provided, and typically includes server, storage and network components. The abstraction layer consists of the software deployed across the physical layer, which manifests the essential cloud characteristics. Conceptually the abstraction layer sits above the physical layer.

languages, libraries, services, and tools supported by the provider.³ The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment.

Infrastructure as a Service (IaaS). The capability provided to the consumer is to provision 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. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., host firewalls).

Deployment Models:

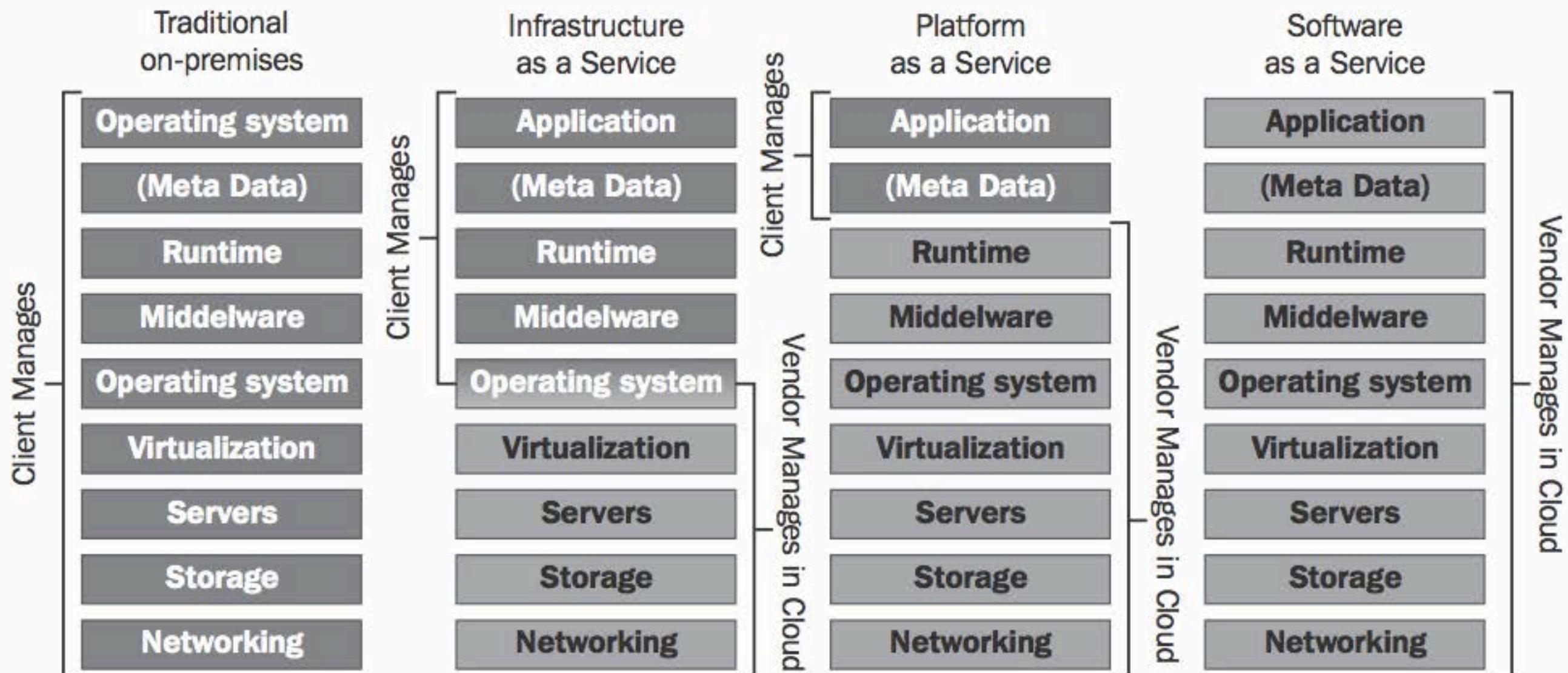
Private cloud. The cloud infrastructure is provisioned for exclusive use by a single organization comprising multiple consumers (e.g., business units). It may be owned, managed, and operated by the organization, a third party, or some combination of them, and it may exist on or off premises.

Community cloud. The cloud infrastructure is provisioned for exclusive use by a specific community of consumers from organizations that have shared concerns (e.g., mission, security requirements, policy, and compliance considerations). It may be owned, managed, and operated by one or more of the organizations in the community, a third party, or some combination of them, and it may exist on or off premises.

Public cloud. The cloud infrastructure is provisioned for open use by the general public. It may be owned, managed, and operated by a business, academic, or government organization, or some combination of them. It exists on the premises of the cloud provider.

Hybrid cloud. The cloud infrastructure is a composition of two or more distinct cloud infrastructures (private, community, or public) that remain unique entities, but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds).

³ This capability does not necessarily preclude the use of compatible programming languages, libraries, services, and tools from other sources.



Customization; higher costs; slower time to value

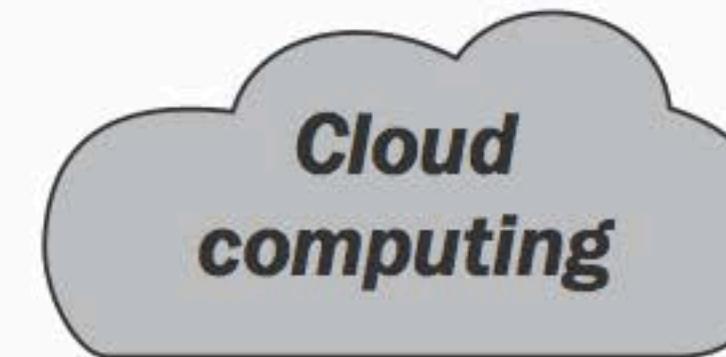
Standardization; lower costs; faster time to value

Characteristics

- On demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service

Automation

Virtualization



Deployment models

- Public Cloud
- Private Cloud
- Hybrid Cloud

Standardization

Service models

- Business Process as a Service
- Software as a Service
- Platform as a Service
- Infrastructure as a Service

An Overview of Cloud Computing

During the last several decades, dramatic advances in computing power, storage, and networking technology have allowed the human race to generate, process, and share increasing amounts of information in dramatically new ways. As new applications of computing technology are developed and introduced, these applications are often used in ways that their designers never envisioned. New applications, in turn, lead to new demands for even more powerful computing infrastructure.

To meet these computing-infrastructure demands, system designers are constantly looking for new system architectures and algorithms to process larger collections of data more quickly than is feasible with today's systems. It is now possible to assemble very large, powerful systems consisting of many small, inexpensive commodity components because computers have become smaller and less expensive, disk drive capacity continues to increase, and networks have gotten faster. Such systems tend to be much less costly than a single, faster machine with comparable capabilities.

Building systems from large numbers of commodity components leads to some significant challenges, however. Because many more computers can be put into a computer room today than was possible even a few years ago, electrical-power consumption, air-conditioning capacity, and equipment weight have all become important considerations for system designs. Software challenges also arise in this environment because writing software that can take full advantage of the aggregate computing power of many machines is far more difficult than writing software for a single, faster machine.

Recently, a number of commercial and academic organizations have built large systems from commodity computers, disks, and networks, and have created software to make this hardware easier to program and manage. These organizations have taken a variety of novel approaches to address the challenges outlined above. In some cases, these organizations have used their hardware and software to provide storage, computational, and data management services to their own internal users, or to provide these services to external customers for a fee. We refer to the hardware and software environment that implements this service-based environment as a *cloud-computing* environment. Because the term "cloud computing" is relatively new, there is not universal agreement on this definition. Some people use the terms *grid computing*, *utility computing*, or *application service providers* to describe the same storage, computation, and data-management ideas that constitute cloud computing.

Regardless of the exact definition used, numerous companies and research organizations are applying cloud-computing concepts to their business or research problems including Google, Amazon, Yahoo, and numerous universities. This article provides an overview of some of the most popular cloud-computing services and architectures in use today. We also describe potential applications for cloud computing and conclude by discussing areas for further research.



Image credit: NOAA Photo Library, NOAA Central Library; OAR/ERL/National Severe Storms Laboratory (NSSL)



Nomenclature

Before describing examples of cloud computing technology, we must first define a few related terms more precisely. A computing *cluster* consists of a collection of similar or identical machines that physically sit in the same computer room or building. Each machine in the cluster is a complete computer consisting of one or more CPUs, memory, disk drives, and network interfaces. The machines are networked together via one or more high-speed local-area networks. Another important characteristic of a cluster is that it's owned and operated by a single administrative entity such as a research center or a company. Finally, the software used to program and manage clusters should give users the illusion that they're interacting with a single large computer when in reality the cluster may consist of hundreds or thousands of individual machines. Clusters are typically used for scientific or commercial applications that can be parallelized. Since clusters can be built out of commodity components, they are often less expensive to construct and operate than supercomputers.

Although the term *grid* is sometimes used interchangeably with cluster, a computational grid takes a somewhat different approach to high-performance computing. A grid typically consists of a collection of heterogeneous machines that are geographically distributed. As with a cluster, each machine is a complete computer, and the machines are connected via high-speed networks. Because a grid is geographically distributed, some of the machines are connected via wide-area networks that may have less bandwidth and/or higher latency than machines sitting in the same computer room. Another important distinction between a grid and a cluster is that the machines that constitute a grid may not all be owned by the same administrative entity. Consequently, grids typically provide services to authenticate and authorize users to access resources on a remote set of machines on the same grid. Because researchers in the physical sciences often use grids to collect, process, and disseminate data, grid software provides services to perform bulk transfers of large files between sites. Since a computation may involve moving data between sites and performing different computations on the data, grids usually provide mechanisms for managing long-running jobs across all of the machines in the grid.

Grid computing and cluster computing are not mutually exclusive. Some high-performance computing systems combine some of the attributes of both. For example, the Globus Toolkit [1], a set of software tools that is currently the *de facto* standard for building grid-computing systems, provides mechanisms to manage clusters at different sites that are part of the same grid. As you'll see later in this article, many cloud-computing systems also share many of the same attributes as clusters and grids.

The Google Approach to Cloud Computing

Google is well known for its expanding list of services including their very popular search engine, email service, mapping services, and productivity applications. Underlying these applications is Google's internally developed cloud-based computing infrastructure. Google has published a series of papers in the computer-science research literature that demonstrate how they put together a small collection of good ideas to build a wide variety of high performance, scalable applications. In this section we describe what Google has built and how they use it.

Google Design Philosophy

Although Google's clouds are very high-performance computer systems, the company took a dramatically different approach to building them than what is commonly done today in the high-performance and enterprise-computing communities [2]. Rather than building a system from a moderate number of very high-performance computers, Google builds their cloud hardware as a cluster containing a much larger number of commodity computers. They assume that hardware will fail regularly and design their software to deal with that fact. Since Google is not using state-of-the-art

hardware, they’re also not using the most expensive hardware. Consequently, they can optimize their costs, power consumption, and space needs by making appropriate tradeoffs.

Another key aspect of Google’s design philosophy is to optimize their system software for the specific applications they plan to build on it. In contrast, the designers of most system software (e.g. operating systems, compilers, and database management systems) try to provide a combination of good performance and scalability to a wide user base. Since it is not known how different applications will use system resources, the designer uses his or her best judgment and experience to build systems that provide good overall performance for the types of applications they expect to be run most often. Because Google is building both the system and application software, they know what their applications require and can focus their system-software design efforts on meeting exactly those requirements.

Google File System

Google’s design philosophy is readily evident in the architecture of the Google File System (GFS) [3]. GFS serves as the foundation of Google’s cloud software stack and is intended to resemble a Unix-like file system to its users. Unlike Unix or Windows file systems, GFS is optimized for storing very large files (> 1 GB) because Google’s applications typically manipulate files of this size. One way that Google implements this optimization is by changing the smallest unit of allocated storage in the file system from the 8 KB block size typical of most Unix file systems to 64 MB. Using a 64 MB block size (Google calls these blocks *chunks*) results in much higher performance for large files at the expense of very inefficient storage utilization for files that are substantially smaller than 64 MB.

Another important design choice Google makes in GFS is to optimize the performance of the types of I/O access patterns they expect to see most frequently. A typical file system is designed to support both sequential and random reads and writes reasonably efficiently. Because Google’s applications typically write a file sequentially once and then only read it, GFS is optimized to support append operations. While any portion of a file may be written in any order, this type of random write operation will be much slower than operations to append new data to the end of a file.

Since GFS is optimized for storing very large files, Google designed it so that the chunks that constitute a single GFS file do not need to reside on one disk as they would in a conventional Unix file system. Instead, GFS allocates chunks across all of the machines in the cloud. Doing chunk allocation in this manner also provides the architectural underpinnings for GFS fault tolerance. Each chunk can be replicated onto one or more machines in the cloud so that no files are lost if a single host or disk drive fails. Google states that they normally use a replication factor of three (each chunk stored on three different machines), and that they do not use the fault-tolerance techniques used in enterprise-class servers such as redundant arrays of inexpensive disks (RAID).

MapReduce

Built on top of GFS, Google’s MapReduce framework is the heart of the computational model for their approach to cloud computing [4, 5]. The basic idea behind Google’s computational model is that a software developer writes a program containing two simple functions—*map* and *reduce*—to process a collection of data. Google’s underlying runtime system then divides the program into many small tasks, which are then run on hundreds or thousands of hosts in the cloud. The runtime system also ensures that the correct subset of data is delivered to each task.

The developer-written *map* function takes as its input a sequence of $\langle key_{in}, value_{in} \rangle$ tuples, performs some computation on these tuples, and produces as its output a sequence of $\langle key_{out}, value_{out} \rangle$ tuples. There does not necessarily need to be a one-to-one correspondence between input and output key/value tuples. Also, key_{in} does not necessarily equal key_{out} for a given key/value tuple.

The developer-written *reduce* function takes as its input a key and a set of values corresponding to that key. Thus, for all $\langle key', value_i \rangle$ tuples produced by the *map* function that have the same key key' , the *reduce* function will be invoked once with key' and the set of all values $value_i$. It’s important to note that if the tuple $\langle key', value_i \rangle$ is generated multiple times by the *map* function, $value_i$ will appear the same number of times in the set of values provided to the *reduce* function, i.e., duplicate values are *not* removed. Once invoked, the *reduce* function will perform some computation on the set of values and produce some output value that the runtime

infrastructure will associate with the key that was supplied as input to *reduce*.

Example

To illustrate how MapReduce might be used to solve a real problem, consider the following hypothetical application. Suppose a software developer is asked to build a tool to search for words or phrases in a collection of thousands or millions of text documents. One common data structure that is useful for building this application is an *inverted index*. For every word w that appears in any of the documents, there will be a record in the inverted index that lists every document where w appears at least once. Once the inverted index is constructed, the search tool can rapidly identify where words appear in the document collection by searching the index rather than the entire collection.

Constructing the inverted index is straightforward using MapReduce. To do so, the developer would construct a *map* function that takes as its input a single $\langle\text{document name}, \text{document contents}\rangle$ tuple, parses the document, and outputs a list of $\langle\text{word}, \text{document name}\rangle$ tuples. For one invocation of *map*, key_{in} might be “*speech.txt*”, and value_{in} might be “*We choose to go to the moon in this decade and do the other things—not because they are easy, but because they are hard!*” If the *map* function were invoked with the key/value tuple shown above, *map* would parse the document by locating each word in the document using whitespace and punctuation, removing the punctuation, and normalizing the capitalization. For each word found, *map* would output a tuple $\langle\text{key}_{out}, \text{value}_{out}\rangle$ where key_{out} is a word and value_{out} is the name of the document key_{in} . In this example, 25 $\langle\text{key}_{out}, \text{value}_{out}\rangle$ tuples would be output as follows

```
<we, speech.txt>
<choose, speech.txt>
<to, speech.txt>
<go, speech.txt>
<to, speech.txt>
...
<hard, speech.txt>
```

The *map* function would be invoked for each document in the collection. If the cloud has thousands of nodes, *map* could be processing

thousands of documents in parallel. Accessing any document from any machine in the cloud via GFS makes *map* task scheduling easier, since a file doesn’t need to be pre-positioned at the machine processing it.

The *reduce* function in this example is very easy to implement. The MapReduce infrastructure will aggregate all $\langle\text{key}, \text{value}\rangle$ tuples with the same key that were generated by all map functions, and send the aggregate to a single *reduce* function as $\langle\text{key}, \text{list of values}\rangle$. This *reduce* invocation will iterate over all values and output the key and all values associated with it. In our example, suppose we have a second document whose name is “*song.txt*” and its contents are “*I’ll see you on the dark side of the moon.*” The map functions processing *speech.txt* and *song.txt* will output the tuples $\langle\text{moon}, \text{speech.txt}\rangle$ and $\langle\text{moon}, \text{song.txt}\rangle$ respectively. *Reduce* will be invoked with the key/value tuple $\langle\text{moon}, \text{list}(\text{speech.txt, song.txt})\rangle$ and will output something that looks like

```
moon: speech.txt song.txt
```

If the word *moon* appeared in other documents, those document names would also appear on this line.

Hundreds or thousands of *reduce* functions will process the output of different *map* functions the same way, once again taking advantage of the parallelism in the cloud. Because of the way the MapReduce infrastructure allocates work, the processing of a single word w will be done by only one *reduce* task. This design decision dramatically simplifies scheduling *reduce* tasks and improves fault tolerance since a failed *reduce* task can be restarted without affecting other *reduce* tasks and without doing unnecessary work.

After all *reduce* functions have finished processing tuples, there will be a collection of files containing one or more lines of *word: document list* mappings as shown above. By concatenating these files, we have constructed the inverted index.

Discussion

The MapReduce model is interesting from a number of perspectives. Decades of high-performance-computing experience has demonstrated the difficulty of writing software that takes full

advantage of the processing power provided by a parallel or distributed system. Nevertheless, the MapReduce runtime system is able to automatically partition a computation and run the individual parts on many different machines, achieving substantial performance improvements. Part of the reason that this is possible goes back to Google's design philosophy. MapReduce is not a general-purpose parallel programming paradigm, but rather a special-purpose paradigm that is designed for problems that can be partitioned in such a way that there are very few dependencies between the output of one part of the computation and the input of another. Not all problems can be partitioned in this manner, but since many of Google's problems can, MapReduce is a very effective approach for them. By optimizing MapReduce performance on their cloud hardware, Google can amortize the costs and reap the benefits across many applications.

Another key benefit of the special-purpose nature of MapReduce is that it can be used to enhance the fault-tolerance of applications. Recall that Google builds its clouds as clusters of commodity hardware and designs its software to cope with hardware failures. Because a MapReduce computation can be partitioned into many independent parts, the MapReduce runtime system can restart one part of the computation if its underlying hardware fails. This restarting operation can be accomplished without affecting the rest of the computation, and without requiring additional expertise or programming effort on the part of the application developer. Google's approach to fault tolerance is in stark contrast to most approaches today that require substantial programmer effort and/or expensive hardware redundancy.

Performance

How well does MapReduce work in practice? Google's published results show impressive results for processing large data. In a paper published in 2004 [4], Google describes two experiments they ran on an 1800 machine cloud. Each machine had two 2 GHz Intel Xeon processors, 4 GB of memory, two 160 GB disks connected via an IDE interface, and gigabit Ethernet. Although not explicitly stated in the paper, we assume the processors were single-core processors.

Google's first experiment was to create and run a program they called grep, which was designed to search for a three-character pattern in 10^{10}

100-byte records—roughly one terabyte of data. Google states the pattern only appeared in 92,337 of the records. They claim their MapReduce implementation of grep was able to find the pattern in all of the data in approximately 150 seconds.

Their second experiment was modeled after the TeraSort benchmark [6] and was designed to sort a terabyte of data (10^{10} 100-byte records). The MapReduce sort implementation was able to sort this data in 891 seconds, which was faster than the fastest reported TeraSort benchmark at that time (1,057 seconds). This result is also interesting from a software-engineering perspective. Google claims that it took less than 50 lines of user code to implement this sort program. Because most of the details of task scheduling and file management are hidden in the MapReduce and GFS runtime system, it is possible to write such a simple program.

In a later paper on MapReduce [5], Google quantifies how many MapReduce jobs are run and how much data is processed in their production systems. They claim that in September 2007 their production MapReduce clouds processed over 400 *petabytes* of data in 2,217,000 jobs (one petabyte equals 1,000 terabytes, or 10^{15} bytes). These numbers are very impressive and demonstrate how useful MapReduce has become for Google's production computing environment.

Bigtable

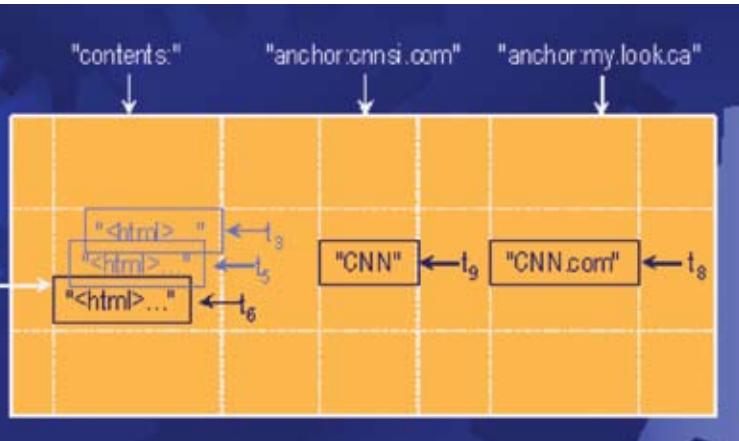
The last major component of Google's approach to cloud computing is their Bigtable data storage system [7]. In many respects, Bigtable superficially resembles a relational database management system (RDBMS). Both store data in tabular form with labeled rows and columns, and they allow data to be searched using the row name (and possibly the column name) as keys. Both also allow new data to be added, data in existing rows to be updated, and data to be deleted. Despite these similarities, there are also some important differences that we shall describe.

Consistent with their design philosophy, Google designed Bigtable so that it performs well in areas most relevant to Google's applications at the expense of generality and performance in less relevant areas. One area where Google paid particular attention was to design Bigtable so that it could store very large quantities of data. Bigtable horizontally partitions an individual table by grouping sets of rows into

tablets, which are managed by different machines in a Google cloud. Since Bigtable stores table rows in lexicographic order, the developer can assign row keys in a manner that increases data locality. Google claims that Bigtable can store petabytes of data across thousands of servers [7]. In this same paper, Google reported that their Google Maps imagery data consumed 70 TB of space in Bigtable. They also reported that their Crawl Project (not described in the paper) data consumed 800 TB of space in a Bigtable table. The Google Maps and Crawl Project data are from 2006, so we assume that larger tables are being stored in Bigtable today.

Bigtable expands on the concepts of rows and columns from relational databases in a couple of ways. In Bigtable an individual data record, called a *cell*, is referenced by a row name, a column name, and a *timestamp*. The timestamp allows multiple versions of the same data to be stored in the same cell. Bigtable also extends the idea of a column by introducing the concept of *column families*. A column family is a set of related columns. Every column belongs to one column family, but a column family can have as many columns as desired. Google designed Bigtable so that creating a new column family is a relatively heavyweight operation while adding or deleting columns to/from a column family is a very lightweight operation. In contrast, changing the schema of a relational database is typically a time-consuming operation that requires shutting down the database during the change.

Figure 1: BIGTABLE example



Source: Chang F, Dean J, Ghemawat S, Hsieh WC, Wallach DA, Burrows M, Chandra T, Fikes A, Gruber RE. Bigtable: A distributed storage system for structured data. ACM Trans. on Computer Systems. 2008 June; 26(2).

Since Bigtable makes it easy to add columns to a column family, and columns can be sparsely populated, data can be stored in ways that would be inefficient or slow in a conventional relational database. Google describes an example Bigtable table (see Figure 1) that stores the contents of specific web pages as well as information about how other web pages link to them [7]. Each row in this table represents a specific web page. The row name is the URL of the page written in reverse order (`com.cnn.www` in the example) to improve locality of reference for related URLs stored in the table. Using the timestamp feature of Bigtable, multiple snapshots of the web page are stored at different points in time in the `contents:` column family. This table also has a column family called `anchor:`, which contains a column for every web page that points to the URL specified in the row name. The column name is the URL of the web page where the link was found. Every cell in the `anchor:` columns contains the “anchor” text whose underlying link is the URL of the row name. In the example, the `anchor:` column family has a column `cnnsi.com` because there is a link on the `cnnsi.com` web page to the `www.cnn.com` web page. The text “CNN” is stored in the corresponding cell, because that is the text on the `cnnsi.com` website that, when clicked in a web browser by a user, causes the browser to go to the page `www.cnn.com`. Thus, every time a new web page is discovered that points to `www.cnn.com`, a new column is added to the `anchor:` family. The name of this column is the new website’s URL, and the cell referenced by this column and the `www.cnn.com` row contains the appropriate anchor text. Although it would be possible to build a relational database to store the data in this table, the cost of frequently adding new columns and the inefficiency of leaving most of the cells in them empty would make doing so impractical.

To implement the functionality in Bigtable that is most important to Google’s applications, Google also omitted some functionality that would normally be found in an RDBMS. For example, Bigtable does not implement a join operation. Another database concept that Bigtable lacks is the transaction. While an individual row in a table can be updated atomically, operations across rows cannot be performed atomically or rolled back. For Google’s applications, these omissions are not a serious impediment. Instead, they simplify the design

of other parts of Bigtable. For example, deleting a column or a column family in a table becomes more complex if there is a running transaction that is manipulating rows with data in affected columns or column families.

Hadoop

After Google published the series of scientific papers describing their approach to cloud computing and their successful experiences using it [2, 5, 7], the approach generated a great deal of interest and enthusiasm outside of Google. Using insights gained from these papers, the open source software development community has created an implementation of Google's cloud approach called Hadoop [8]. Hadoop is part of the Apache open source project and contains an implementation of MapReduce and a GFS-like distributed file system called Hadoop Distributed File System (HDFS). HBase, a related Apache project, is an open source implementation of Bigtable [9].

As with GFS, HDFS stores large files across large clusters in sequences of blocks. Replication is also included within HDFS. Hadoop MapReduce and HDFS, as well as HBase, employ master/slave architectures very similar to the approach Google took in designing its corresponding systems. Unlike Google's systems, which are written in C++, Hadoop and HBase are written in Java.

Hadoop began as part of the Nutch open source search engine in 2002-2004 [10]. Nutch was built by a small group of developers working part-time. After the 2003-2004 publication of Google's papers, work began to add some of the Google concepts to Nutch, in order to address some of Nutch's scalability limitations. By 2006, Hadoop was split out of Nutch and became a separate effort. Since its launch, the community using and supporting Hadoop has grown substantially. Hadoop is currently in release 0.19.1, and there are now many Hadoop user groups and applications. Yahoo is currently a major supporter of Hadoop. To give a sense of how far Hadoop has matured in a short time, a 900-node Hadoop cluster at Yahoo set a new TeraSort benchmark in July 2008 [11]. This cloud was able to sort a terabyte of data in 209 seconds, beating the old record of 297 seconds. According to Yahoo, this is the first time a Java implementation or an open source TeraSort implementation has set a record for this benchmark.

HBase is newer and less mature than Hadoop. As with Hadoop, a Google scientific paper motivated its development. Initial work on HBase began in 2006 and was first released with Hadoop 0.15 in October 2007.

The Google approach to cloud computing is also gaining interest in academia. Google has joined forces with IBM to initiate university research to address large-scale computing problems across the Internet [12]. The current initiative is with Carnegie Mellon University, the University of Maryland, the Massachusetts Institute of Technology, Stanford University, and the University of California, Berkeley. Google and IBM are providing computing hardware to these universities to run the Hadoop and HBase software.

Professor Jimmy Lin, a faculty member in the College of Information Studies at the University of Maryland, College Park, brought together students from the Computational Linguistics and Information Processing Laboratory and University of Maryland Institute for Advanced Computer Studies to examine large-scale natural-language processing problems using Hadoop. This special seminar course was first held in the Spring 2008 with five projects. PhD students led small teams to explore open research problems and used MapReduce to assist with large scale parallelization issues associated with statistical machine translation, language modeling, identity resolution in email, biomedical network analysis, and image processing. Included in the course were various speakers who led discussions on cloud computing during the semester.

Amazon Approach to Cloud Computing

Amazon is best known for selling books online, but they are also actively investing in services that allow developers to take advantage of their computing technology. Amazon Web Services provide developers use of open APIs to access Amazon's vast infrastructure in a manner vaguely reminiscent of timeshared computing. By using these APIs, developers can create interfaces and access the computing infrastructure provided by Amazon on a fee-based schedule, with the ability to grow as needed. Software developers, start-up companies, and established companies in need of reliable computing power are members of a large and growing crowd using Amazon services.

One of these services is the beta launch of Amazon Elastic Compute Cloud or EC2 [13]. The Amazon Elastic Compute Cloud provides virtualization for developers to load Amazon-managed machines with their preferred software environments and execute custom applications. This is accomplished by first creating an Amazon Machine Instance (AMI) with the operating system, custom configuration settings, libraries, and all needed applications. Once created, the AMI is loaded into the Amazon Simple Storage Service (AS3) and receives a unique identifier. The unique identifier can then be used to run as many instances of the AMI as needed using Amazon's APIs. Additionally, Amazon provides a set of pre-built AMIs that can be used by developers.

AMIs can be sized to the requirements of individual applications. AMIs fall into categories ranging from a small instance to an extra-large instance. A small instance has less memory, virtual cores, storage, and I/O performance than a large one. Similar to a timesharing system, Amazon bills users by the instance-hour. As the size of memory, number of cores, or other features increases, the instance-hour fee increases. Amazon offers standard instances as well as high-CPU instances.

Amazon is also now claiming location transparency for a globally distributed cloud. They are building out their computational footprint to be more geographically distributed. Additionally, they are improving fault tolerance by creating Availability Zones that will allow users to create instances of their applications in distributed regions.

Microsoft Approach to Cloud Computing

Microsoft announced its Azure Services Platform in October 2008 [14, 15]. Similar to the Amazon approach, Microsoft is developing a cloud-based, hosted-services platform. In addition to providing compute and storage resources for consumers to develop and host applications, Microsoft is also offering cloud applications that are already developed and ready for consumption.

The Azure Service Platform is built on the Windows Azure cloud operating system, which provides a development, hosting, and management environment for cloud applications. Numerous services are available on top of the Azure operating system including Live Services, SQL Services and .NET Services.

During the Community Technology Preview, Azure is offered for free to allow users and consumers to test and evaluate it. Potential users can also download an Azure SDK and Azure tools for Microsoft Visual Studio to simulate the Azure framework during the preview period. Once Azure is launched for commercial use it will be priced using a consumption-based model. Consumption will be measured in compute time, bandwidth, and storage and transactions (put and gets).

Microsoft is using a combination of Microsoft .NET framework and the Microsoft Visual Studio development tools to provide a base for developers to easily launch new solutions in the cloud. It is noted that both applications running in the cloud and outside of the cloud can use the Azure cloud platform. For the initial offering, only applications built with .NET can be hosted, but Microsoft claims that this constraint will be relaxed for Azure in 2009.

Azure's storage framework is based on storing of binary large objects (blobs), communications queues to provide access to the data via Azure applications, and a query language that can provide table-like structures. An Azure account holder can have one or more containers where each container can hold one or more blobs. Each blob has a maximum size of 50 GB, and can be subdivided into smaller blocks. To work with the blobs of data, entity and property hierarchies are provided through tables. These tables are not SQL-like relational tables and are not accessed using SQL. Instead, access to these tables is provided via the Microsoft Language Integrated Query (LINQ) syntax query language. Queues are also available to provide communication between instances as will be discussed later. Representational State Translation (REST) is the convention used to both expose and identify data stored in the Azure cloud. All Azure data storage is replicated three times to enhance fault tolerance.

The .NET Services provide access control at the application level, a service bus for exposing application services and allowing services to communicate with each other, and a workflow management system for creating complex services from existing simpler services (service orchestration).

SQL Services will be used to provide a set of services for working with both unstructured and relational data. The first set of SQL Services will only provide database services in the Azure cloud,

but will expand in the future. While this service appears similar to a relational database storage model, Microsoft claims that it is slightly different in that it is a hierarchical data model without a pre-defined schema. Therefore, access to this data is not provided via a structure query language but instead through the RESTful interface or C# syntax defined by Microsoft's LINQ.

The Live Services use the cloud to store data while running on a variety of live operating desktop and mobile systems. This allows the Live Operating System to synchronize data across numerous related mesh systems. As an example, a user can create a mesh with his/her desktop, laptop, and mobile phone and keep them seamlessly synchronized at all times.

Windows Azure divides application instances into virtual machines (VMs) similar to the Amazon AMIs described earlier. Unlike the Amazon AMIs, a Windows Azure developer cannot supply his/her own VM image. Developers create Web role and Worker role instances, where a Web role accepts incoming HTTP requests and Worker roles are triggered by Web roles via a queue. Any work performed by a Worker role instance can then be stored in the Azure storage or sent outside of the cloud network. Web role instances are stateless. To expand the performance of an application, multiple Worker role instances can be run across dedicated processor cores. If a Worker role or Web role fails, the Azure fabric restarts it.

Other Cloud Computing Approaches and Applications

Amazon, Google, and Microsoft are not alone investing in computing as a service. Other organizations to test the waters include Dell, IBM, Oracle, and some universities.

IBM is providing a variety of cloud-based services by using existing functionality and capabilities of the IBM Tivoli portfolio [16]. Tivoli is a collection of products and software services that can be used as building blocks to support IBM Service Management software. IBM's cloud-based services, which target independent software vendors (ISVs), offer design of cloud infrastructures, use of worldwide cloud computing centers, and integration of cloud services.

Researchers at the University of Michigan (UM) have developed a novel anti-virus application using

cloud computing ideas [17]. By aggregating a collection of open source and commercial anti-virus software as a cloud-based service and letting the individual anti-virus packages "vote" on whether an infection has occurred on a host, they demonstrated that their CloudAV service was more effective at detecting viruses than any single anti-virus software package. With a small software client running on each end user host, the UM researchers also claimed that a centralized virus detection system would be easier to manage in an enterprise than maintaining signature files and software releases on hundreds or thousands of end hosts.

Concerns and Challenges

Perhaps the biggest danger that arises when a technology gains sufficient interest from enough people is that it will begin to be viewed as a panacea. Gartner refers to such a situation as the *peak of inflated expectations* in their Hype Cycle [18]. While we believe that cloud computing can indeed be applied to many kinds of problems successfully, we also think that it's necessary to consider carefully whether the problem needing to be solved could best be addressed by an existing technology.

When we described Google's Bigtable data storage system, we compared it to RDBMSs. There are many problems that are best solved using a relational database, and systems like Bigtable do not add value. For example, a fundamental requirement of a banking database is that information about how much money is in each customer's bank account must be accurate at all times, even while money is being transferred between accounts or after a system has crashed. Such an application cries out for a transactional model that is part of an RDBMS, but not Bigtable. Being able to store petabytes of data is less important here than being able to execute transactions correctly.

The Amazon approach to cloud computing is ideal for small organizations or organizations with unpredictable computing usage requirements. For large organizations or organizations that process particularly sensitive data, this approach may not

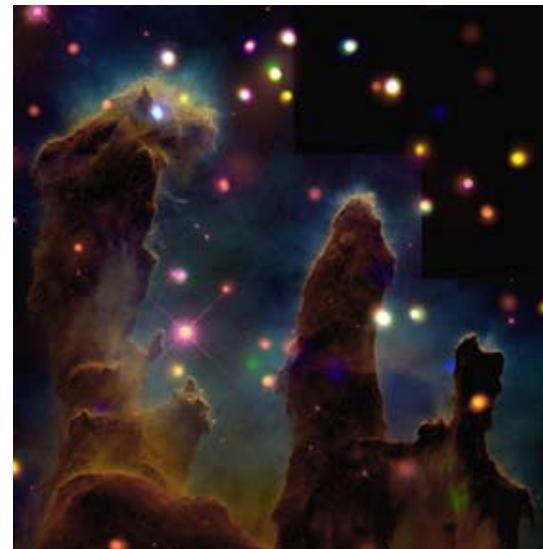


Image credit:
X-ray: NASA/CXC/U.
Colorado/Linsky et al.;
Optical: NASA/ESA/STScI/
ASU/J.Hester & P.Scowen.



Image credit: NOAA Photo Library, NOAA Central Library; OAR/ERL/National Severe Storms Laboratory (NSSL)

make sense. With the Amazon approach, a user is effectively renting computing resources. Renting computing resources may not be the most cost-effective use of funds for a large corporation. As an organization grows in size and importance, the *value* of its data also increases dramatically. An automotive manufacturer would probably not want to store the highly proprietary designs for next year's car models on another company's servers. Similarly, a government agency and the citizens it serves would probably not want sensitive data such as citizens' tax returns to be stored on a computer system that is not owned and controlled by the government.

Cloud computing approaches use parallelism to improve the computational performance of applications. The Google MapReduce framework is particularly good at this so long as the problem fits the framework. Other approaches to high performance computing have similar constraints. It's very important for developers to understand the underlying algorithms in their software and then match the algorithms to the right framework. If the software is single-threaded, it will not run faster on a cloud, or even on a single computer with multiple processing cores, unless the software is modified to take advantage of the additional processing power. Along these lines, some problems cannot be easily broken up into pieces that can run independently on many machines. Only with a good understanding of their application and various computing frameworks can developers make sensible design decisions and framework selections.

Future Research Areas

Although much progress has already been made in cloud computing, we believe there are a number of research areas that still need to be explored. Issues of security, reliability, and performance should be addressed to meet the specific requirements of different organizations, infrastructures, and functions.

Security

As different users store more of their own data in a cloud, being able to ensure that one user's private data is not accessible to other users who are not authorized to see it becomes more important. While virtualization technology offers one approach for improving security, a more fine-grained approach would be useful for many applications.

Reliability

As more users come to depend on the services offered by a cloud, reliability becomes increasingly important, especially for long-running or mission-critical applications. A cloud should be able to continue to run in the presence of hardware and software faults. Google has developed an approach that works well using commodity hardware and their own software. Other applications might require more stringent reliability that would be better served by a combination of more robust hardware and/or software-based fault-tolerance techniques.

Vulnerability to Attacks

If a cloud is providing compute and storage services over the Internet such as the Amazon approach, security and reliability capabilities must be extended to deal with malicious attempts to access other users' files and/or to deny service to legitimate users. Being able to prevent, detect, and recover from such attacks will become increasingly important as more people and organizations use cloud computing for critical applications.

Cluster Distribution

Most of today's approaches to cloud computing are built on clusters running in a single data center. Some organizations have multiple clusters in multiple data centers, but these clusters typically operate as isolated systems. A cloud software architecture that could make multiple geographically distributed clusters appear to users as a single large cloud would provide opportunities to share data and perform even more complex computations than

possible today. Such a cloud, which would share many of the same characteristics as a grid, could be much easier to program, use, and manage than today's grids.

Network Optimization

Whether clouds consist of thousands of nodes in a computer room or hundreds of thousands of nodes across a continent, optimizing the underlying network to maximize cloud performance is critical. With the right kinds of routing algorithms and Layer 2 protocol optimizations, it may become possible for a network to adapt to the specific needs of the cloud application(s) running on it. If application-level concepts such as locality of reference could be coupled with network-level concepts such as multicast or routing algorithms, clouds may be able to run applications substantially faster than they do today. By understanding how running cloud applications affects the underlying network, networks could be engineered to minimize or eliminate congestion and reduce latency that would degrade the performance of cloud-applications and non-cloud applications sharing the same network.

Interoperability

Interoperability among different approaches to cloud computing is an equally important area to be studied. There are many cloud approaches being pursued right now and none of them are suitable for all applications. If every application were run on the most appropriate type of cloud, it would be useful to share data with other applications running on other types of clouds. Addressing this problem may require the development of interoperability standards. While standards may not be critical during the early evolution of cloud computing, they will become increasingly important as the field matures.

Applications

Even if all of these research areas could be addressed satisfactorily, one important challenge remains. No information technology will be useful unless it enables new applications, or dramatically improves the way existing applications are built or run. Although the effectiveness of cloud computing has already been demonstrated for some applications, more work should be done on identifying new classes of novel applications that can only be realized using cloud computing technology. With proper instrumentation of potential applications

and the underlying cloud infrastructure, it should be possible to *quantitatively* evaluate how well these application classes perform in a cloud environment. Along these same lines, experimental software engineering research should be conducted to measure how easily new cloud-based applications can be constructed relative to non-cloud applications that perform similar functions. This research should also compare the dependability of similar cloud and non-cloud based applications running in production environments. Application-focused research will help organizations make well-informed business decisions on where to apply cloud technology, and give cloud technology developers guidance on what kinds of improvements to the technology will provide the greatest benefits to application developers and end users.

Conclusion

We have described a number of approaches to cloud computing in this article and pointed out some of their strengths and limitations. We have also provided motivation and suggestions for additional research. The approaches outlined in this article, along with other strategies, have already been applied successfully to a wide range of problems. As more experience is gained with cloud computing, the breadth and depth of cloud implementations and the range of application areas will continue to increase.

Like other approaches to high performance computing, cloud computing is providing the technological underpinnings for new ways to collect, process, and store massive amounts of information. Based on what has been demonstrated thus far, ongoing research efforts, and the continuing advancements of computing and networking technology, we believe that cloud computing is poised to have a major impact on our society's data-centric commercial and scientific endeavors. ■

References and Further Reading:

- [1] Globus Alliance homepage. <http://www.globus.org>, 2008.
- [2] Barroso LA, Dean J, Hölzle U. Web search for a planet: the Google cluster architecture. *IEEE Micro*. 2003 March-April; 23(2).
- [3] Ghemawat S, Gobioff H, Leung S. The Google filesystem. *Proceedings ACM Symposium on Operating Systems Principles*. 2003 October.
- [4] Dean J and Ghemaway S. MapReduce: Simplified data processing on large clusters. *Proceedings Operating Systems Design and Implementation*. 2004 December.
- [5] Dean J and Ghemaway S. MapReduce: Simplified data processing on large clusters. *Communications of the ACM*. 2008 January; 51(1).
- [6] Gray J. Sort Benchmark home page. <http://research.microsoft.com/barc/SortBenchmark>, 2006.
- [7] Chang F, Dean J, Ghemawat S, Hsieh WC, Wallach DA, Burrows M, Chandra T, Fikes A, Gruber RE. Bigtable: A distributed storage system for structured data. *ACM Trans. on Computer Systems*. 2008 June; 26(2).
- [8] Hadoop Project homepage. <http://hadoop.apache.org/core>, 2008.
- [9] HBase Project homepage. <http://hadoop.apache.org/hbase>, 2008.
- [10] Cutting D. Hadoop: a brief history [Internet]. <http://research.yahoo.com/files/cutting.pdf>, 2008.
- [11] Yahoo Developer Blogs. Apache Hadoop wins terabyte sort benchmark [Internet]. http://developer.yahoo.com/blogs/hadoop/2008/07/apache_hadoop_wins_terabyte_sort_benchmark.html, 2008.
- [12] IBM homepage. IBM and Google announce university initiative to address internet-scale computing challenges [Internet]. <http://www-03.ibm.com/press/us/en/pressrelease/22414.wss>, 2007.
- [13] Amazon Web Services. Amazon Elastic Compute Cloud homepage. <http://aws.amazon.com/ec2>, November 2008.
- [14] Chappel D. Microsoft Azure homepage. Introducing the Azure Services Platform [Internet]. <http://www.microsoft.com/azure/whatisazure.mspx>, October 2008.
- [15] Microsoft Azure homepage. <http://www.microsoft.com/azure>, 2008.
- [16] IBM homepage. IBM launches cloud services initiative [Internet]. <http://www-03.ibm.com/press/us/en/pressrelease/25341.wss>, 2008.
- [17] Oberheide J, Cooke E, Jahanian F. CloudAV: N-Version antivirus in the network cloud. *Proceedings of the 17th USENIX Security Symposium*; 2008 July.
- [18] Gartner homepage. Understanding Hype Cycles [Internet]. <http://www.gartner.com/pages/story/story.php?id8795.s.8.jsp>, 2008.

The Challenges of Cloud Storage

Given the obvious benefits of Cloud Storage, raises the question as to why haven't more organizations already adopted cloud storage for on-premises applications and alleviate present expensive storage.

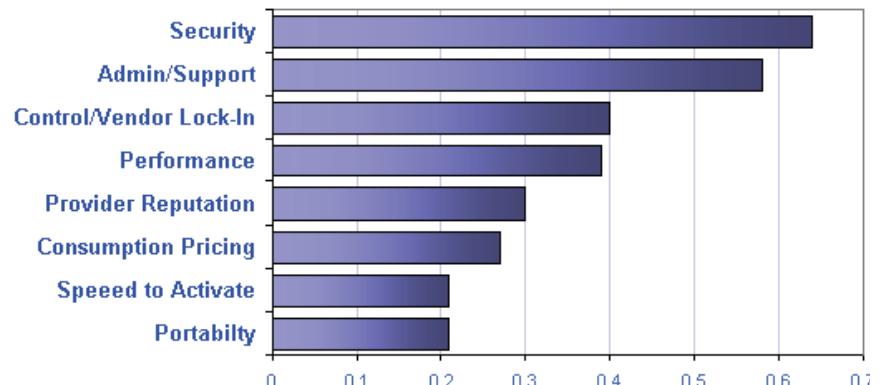
The answer – Right now, there are traditional IT barriers which must be overcome when considering integration of cloud storage into today's production environments.

- Performance and data transfer rates become key issues as the distance between the data and the user increases - which is what happens in cloud computing.
- Even unlimited bandwidth without solving the latency problem will not improve the performance because it is the latency - or the chattiness - of the protocols, plus the speed of light limitations that cause the end user experience to be very poor.
- Not all data access patterns are well suited to the cloud, particularly if there are large distances to cover. In such cases, bandwidth becomes not only a challenge but a financial consideration.
- Cloud storage isn't about to replace the storage network in the data center any time soon, at least not for data-intensive, high-performance, low-response time, transactional applications and for mission-critical data. But we will see many use cases where companies and organizations of all sizes will augment their on-premise storage with cloud storage potentially from various vendors in a hybrid model deployment. However, hybrid models tend to bring interoperability issues and the need to deal with different tools, API's, management infrastructure etc.
- The popular storage use cases tend to be infrequently accessed data scenarios including archiving, backup, DR, and offsite data protection. Performance and data transfer rates become key issues as the distance between the data and the user increases - which is what happens in cloud computing.
- It will be a hybrid cloud storage world for many years to come with a lot of that storage living in private clouds. Many large enterprises deal with petabytes of data and their processing.

A survey conducted by IMEX Research reveals that both enterprises and SMBs today are concerned about:

- **Security & Data Availability** - Concerned about the security of their data, once it is in the hands of the cloud provider whose multi-tenancy infrastructure is shared by others. Numerous questions arise when considering cloud storage, including "is my data secure?", "am I in compliance?", "what happens if a provider loses a disk drive?", "is my data protected?", "how do I know if my data is truly unusable when I delete it?", and many others. StorSimple allows you to provide an encryption key – or we can generate one for you – and all data sent to the cloud is encrypted. They are rightfully worried about the availability of their data and how that impacts their day-to-day operations. What happens if a cloud storage service is offline for a period of time?
- **Performance** - Legitimately concerns about application performance if the application storage is in the cloud. Will the cloud storage service satisfy my workloads?
- **Bandwidth limitations** – Bandwidth is a limiting factor when accessing a public storage cloud, as they are accessed over the Internet. Primary storage deduplication and compression, minimizes bandwidth consumption dramatically while also improving performance.
- **Latency constraints** – Latency is the silent killer of application performance, both in terms of response time and throughput. StorSimple takes advantage of parallelization, persistent connections, and TCP optimizations to overcome latency and improve performance
- **Manageability** - Are concerned about being locked into their proprietary cloud storage infrastructure and applications services. They don't have vendor independent tools or industry standards to evaluate the applicability or measure the effectiveness of cloud storage for their environment.
- **Interoperability/Protocol translation** – A serious concern exists today is: Most of today's on-premises applications use block protocols such as FC, iSCSI etc. But Cloud storage protocols predominantly speak only in the language of file protocols (CIFS, NFS) and both public and private storage clouds are accessed via REST HTTP-based, or SOAP APIs. Since these applications expect block access to storage, introducing a cloud storage system to the application is like trying to have a conversation in Spanish when you only speak English. So how will current applications even be interoperable between existing storage infrastructure and cloud storage? Taking advantage of these conversion tools and seamlessly integrating with cloud storage eliminates the need for complex application re-programming and integration (see below - Importance of Cloud Storage API set).
- **Costs** - Many cloud storage services charge their customers based on the amount of storage capacity consumed and the number of IOPs performed or the amount of bandwidth consumed. Much as reducing cloud storage costs through deduplication and optimization and taking advantage of pay-as-you-grow schemes helps but choosing a cloud storage vendor which meets all other criteria remains a challenge.

Cloud Computing Adoption Obstacles



NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR

Cachar, Assam

B.Tech. VIIIth Semester

Subject Code: CS-484

Subject Name: Cloud Computing

Submitted By:

Name : Subhojit Ghimire

Sch. Id. : 1912160

Branch : CSE – B

1. Suppose, you are managing an identity manager in a company. The company provides identity managers as a service.

a. What are the issues and challenges of such systems?

→ Identity Managers as a Service (IMaaS) are cloud-based solutions that offer identity and access management (IAM) functionalities as a service. They can simplify identity provisioning, authentication, authorization and federation by providing a centralized platform for managing user identities across different cloud services. Some of the issues and challenges of IMaaS are:

1. Security: IMaaS providers must ensure that they protect user data and credentials from unauthorized access, leakage or theft. They must also comply with various regulations and standards regarding data privacy and security.
2. Integration: IMaaS providers must ensure that they support different cloud platforms, devices and applications that their customers use. They must also provide seamless interoperability with existing IAM systems and processes within the customer's organization.
3. Performance: IMaaS providers must ensure that they deliver high availability, scalability and reliability of their services. They must also cope with increasing demand and complexity of IAM tasks and activities.

b. What is Captcha? Can we implement a password-based Captcha? Justify your answer.

→ Captcha is a computer program or system intended to distinguish human from machine input, typically as a way of thwarting spam and automated extraction of data from websites. It is a type of Turing test that challenges users to solve a simple problem that is easy for humans but hard for bots.

A password-based Captcha is a Captcha that requires users to enter a password that is displayed on the screen in a distorted or obscured way. The idea is to prevent bots from guessing or cracking the password by making it difficult for them to read it.

Yes, we can implement a password-based Captcha by generating random passwords and displaying them on the screen with some distortion or noise. For example, we can use reCAPTCHA v2 which offers an option to use text-based challenges instead of images. However, this may not be very effective or secure for the following reasons:

1. Password-based Captchas are vulnerable to optical character recognition (OCR) techniques that can recognize text from images. Bots can use OCR tools or services to bypass the Captcha by reading the password and entering it correctly.
2. Password-based Captchas are also vulnerable to brute-force attacks that can try different combinations of characters until they find the correct password. Bots can use dictionaries or lists of common passwords to speed up this process.

3. Password-based Captchas may not be very user-friendly because they require users to type in long or complex passwords that may be hard to read or remember. This may frustrate users and reduce their satisfaction with the website.

Therefore, a better alternative may be to use reCAPTCHA v3 which does not require any user interaction but returns a score based on how likely the user is human or bot. This score can then be used to decide whether to allow, challenge or block the user.

c. Why cannot we replace a password-based identity management system using an OTP?

- ➔ A password-based identity management system is a system that facilitates a simple, secure way to store passwords and access them quickly when required. It relies heavily on passwords for every service, whether it's something as simple as marking daily attendance or as sensitive as accessing clients' unmasked financial details. On the other hand, an OTP (One-Time Password) is a unique security code used for online transactions. It is an automatically generated numeric string of characters, which acts as a PIN to authenticate various banking transactions. It is valid for only one login session or transaction, on a computer system or other digital device.

We cannot replace a password-based identity management system using an OTP for the following reasons:

1. An OTP is not a replacement for a password but an additional layer of security that complements the password. An OTP alone cannot verify the identity of the user but only the possession of the device that receives the OTP.
2. An OTP is not suitable for every service or transaction that requires authentication. For example, some services may require persistent login sessions that last longer than one-time transactions. An OTP would expire after each session and require users to generate and enter new codes repeatedly.
3. An OTP depends on external factors such as network availability, device battery life, and user accessibility. If any of these factors fail, users may not be able to receive or enter their OTPs and lose access to their accounts or services.

Therefore, a better alternative may be to use multi-factor authentication (MFA) which combines two or more factors such as something users know (password), something users have (OTP), and something users are (biometrics). This way, users can have more security and convenience than using either passwords or OTPs alone.

2. Suppose, you are assigned to build a centralized shopping cart, named OurShop.com, where each shop owner can sell their products, generate revenues and they can brand their own shop. The clients can search their requirements in a single window and be given a search bar. Assume that almost all the shops in India are registered on OurShop.com.

a. What are the key Cloud Computing components of OurShop.com?

→ Some of the key cloud computing components of a centralized shopping cart, OurShop.com, are:

1. Storage: This is where we store our product data, customer data, order data, and other information related to our website. We can use cloud storage services like Amazon S3 or Google Cloud Storage to store our data securely and reliably.
2. Compute: This is where we run our website logic, such as processing orders, calculating prices, generating recommendations, etc. We can use cloud computing services like Amazon EC2 or Google Compute Engine to run our website code on virtual machines or containers that scale according to our demand.
3. Database: This is where we store and query our structured data, such as product catalogue, inventory, customer profiles, etc. We can use cloud database services like Amazon RDS or Google Cloud SQL to manage our relational databases in the cloud.
4. Networking: This is where we connect our website components and enable communication between them. We can use cloud networking services like Amazon VPC or Google Cloud VPC to create private networks for our website resources and control their access and security.
5. Load balancing: This is where we distribute our website traffic across multiple servers or instances to improve performance and availability. We can use cloud load balancing services like Amazon ELB or Google Cloud Load Balancing to balance our traffic automatically and handle spikes in demand.
6. CDN: This is where we deliver our website content faster and more efficiently to our customers around the world. We can use cloud CDN services like Amazon CloudFront or Google Cloud CDN to cache our static content (such as images, videos, etc.) at edge locations near our customers and reduce latency and bandwidth costs.
7. Monitoring: This is where we track and analyse the performance and health of our website components and identify any issues or anomalies. We can use cloud monitoring services like Amazon CloudWatch or Google Cloud Monitoring to collect metrics, logs, events, etc. from our website resources and visualize them on dashboards or alerts.
8. Management: This is where we manage our website components such as storage services, applications, runtime cloud infrastructure, security issues in the backend and establish coordination among them. We can use cloud management services like AWS Management Console or Google Cloud Console to access and control our website resources from a single interface.

9. **Security:** This is where we protect our website data and resources from unauthorized access or attacks. We can use cloud security services like AWS Identity & Access Management (IAM) or Google Cloud IAM to manage user authentication and authorization for accessing our website resources.

b. How do you make your OurShop.com feasible?

→ To make a centralized shopping cart feasible, we need to consider several factors that can affect our website's performance, usability, and profitability. Some of these factors are:

1. **Order summary:** We should provide a clear and concise summary of the items added to the shopping cart, including their quantity, price, subtotal, taxes, shipping costs, and total amount. This helps our customers review their order and make any changes before proceeding to checkout.
2. **Live chat:** We should offer a live chat option for our customers to contact us in real time if they have any questions or issues with their order. This can increase customer satisfaction and trust, as well as reduce cart abandonment rates.
3. **Mobile friendliness:** We should ensure that our shopping cart is responsive and optimized for mobile devices, as more and more customers are using smartphones and tablets to shop online. We should use a readable layout, large buttons, easy navigation, and fast loading times for our mobile shopping cart.
4. **Relevant questions:** We should only ask our customers for the information that is necessary for completing their order, such as their name, email address, shipping address, payment method etc. We should avoid asking irrelevant or optional questions that can distract or annoy our customers and make them abandon their cart.
5. **Payment options:** We should offer multiple payment options for our customers to choose from according to their preference and convenience. We should also integrate with secure payment gateways that can process credit cards or other online payments safely and efficiently.
6. **Trust signals:** We should build trust with our customers by displaying trust signals on our shopping cart page such as security badges (e.g., SSL certificate), customer reviews (e.g., ratings), guarantees (e.g., money-back policy), social proof (e.g., number of orders), etc.

c. What are the unique key components you would like to add to OurShop.com?

→ Following are some of the unique key components that can enhance our centralized shopping cart are:

1. **Large, functional product images:** We should display high-quality and zoomable images of our products that show their features and details clearly. This can help our customers visualize our products better and make informed decisions.

2. Product reviews: We should enable our customers to leave feedback and ratings for our products on our website. This can provide social proof and credibility for our products and increase customer trust and loyalty.
3. Layered and faceted navigation: We should allow our customers to filter and sort our products by various attributes such as price, colour, size, brand, etc. This can help our customers find what they are looking for faster and easier.
4. Single-page, fast checkout: We should simplify our checkout process by reducing the number of steps and pages required to complete an order. This can improve user-friendliness and streamline customer experience, as well as reduce cart abandonment rates.
5. Search: We should provide a powerful and accurate search function on our website that can help our customers find what they need quickly. We should also use autocomplete, spell check, synonyms, etc. to enhance our search results.
6. Coupons and discounts: We should offer incentives such as coupons, discounts, free shipping, etc. to attract more customers and increase sales conversions. We should also make it easy for our customers to apply these offers on our shopping cart page.
7. Universal cart: We should enable our customers to access their shopping cart across multiple channels such as online store, mobile app, physical store etc. This can provide a seamless and consistent shopping experience for our customers regardless of where they shop.

d. How do you attract shopkeepers to sell their products on OurShop.com?

→ To attract shopkeepers to sell their products on our centralized shopping cart, we need to offer them some benefits that can motivate them to join our platform. Some of these benefits are:

1. Increased exposure: We can help shopkeepers reach more customers by listing their products on our website and promoting them through various channels such as Google Merchant Centre, social media, email marketing etc. We can also provide them with analytics and insights on their sales performance and customer behaviour.
2. Reduced costs: We can reduce shopkeepers' operational costs by providing them with a ready-made shopping cart solution that is easy to use and integrate with their existing systems. We can also offer them competitive fees and commissions for using our platform and services.
3. Enhanced customer experience: We can enhance shopkeepers' customer experience by providing them with features such as live chat, mobile-friendly checkout, coupons and discounts, free shipping etc. We can also handle customer service and support issues on behalf of the shopkeepers and ensure customer satisfaction and loyalty.
4. Multipurpose retail spaces: We can encourage shopkeepers to create multipurpose retail spaces that combine online and offline shopping experiences. For example, we can enable shopkeepers to offer in-store pickup or delivery options for online orders or allow customers to order online while browsing in physical stores.

e. What are the hidden issues and challenges of such a system?

→ Some of the hidden issues and challenges of such a system are:

1. Shopping cart abandonment: This is when customers add products to their shopping cart but do not complete their purchase. This can result in lost sales and revenue for our platform. Some of the reasons for shopping cart abandonment are hidden costs, complicated checkout process, lack of trust, etc. We need to optimize our shopping cart page and offer incentives such as free shipping, discounts, etc. to reduce abandonment rates.
2. Cyber and data security: This is when our platform is vulnerable to cyberattacks or data breaches that can compromise our customers' personal and financial information. This can damage our reputation and customer trust. We need to implement strong security measures such as encryption, authentication, firewalls, etc. to protect our platform and data from hackers and malware.
3. Customer experience: This is when our platform fails to meet or exceed our customers' expectations and needs. This can result in low satisfaction and loyalty levels. We need to provide a seamless and fluid experience for our customers across multiple channels such as online store, mobile app, physical store etc. We need to use customer data to personalize our offers and content and deliver relevant recommendations.
4. Competition: This is when our platform faces stiff competition from other similar or superior platforms that offer better products or services. This can reduce our market share and profitability. We need to differentiate ourselves from our competitors by offering unique value propositions such as exclusive products, loyalty programs, social responsibility etc.

f. Discuss the advantages and disadvantages of such a centralised shopping cart.

→ Some of the advantages are:

1. Economy in investments: A centralized shopping cart can reduce the costs of building and maintaining multiple storage facilities and platforms for different shopkeepers. It can also lower the upstream shipping costs when buying products in bulk.
2. Simpler and more efficient management: A centralized shopping cart can make it easier to implement new solutions and technologies, such as smart shopping carts, that can enhance customer experience and satisfaction. It can also provide better product availability and security arrangements to prevent pilferage and theft.
3. Safer shopping: A centralized shopping cart can offer customers a safer shopping option, especially during pandemics or emergencies. Customers can avoid crowded physical stores and shop online from their homes. They can also use contactless payment methods and delivery options.

Some of the disadvantages are:

1. Bureaucratic leadership: A centralized shopping cart can create a dictatorial form of leadership where shopkeepers have little or no say in the decision-making process of the platform. They may have to follow strict rules and regulations that may not suit their needs or preferences. They may also lose their identity and uniqueness as they become part of a larger entity.
2. More material handling operations: A centralized shopping cart can increase the amount of material handling operations that are required to transport products from one location to another. This can result in more time, labour, energy, and risk involved in moving products across long distances.
3. Chances of misunderstanding: A centralized shopping cart can create chances of misunderstanding between shopkeepers who require certain products and customers who order them online. There may be delays or errors in communication or delivery that can affect customer satisfaction and loyalty.

g. Why would such shopping carts be introduced in India?

- Such shopping carts could be introduced in India for various reasons like:
1. Faster buying process: Customers can spend less time shopping for what they want and pay online without having to wait in long queues at the cash counter.
 2. Store and product listing creation: Shop owners can create their own product listings and showcase their products to a large number of customers across India.
 3. Cost reduction: Shop owners can save on rent, utilities, staff, and other expenses by selling online instead of having a physical store.
 4. Affordable advertising and marketing: Shop owners can use online platforms and social media to promote their products and reach more potential customers.
 5. Flexibility for customers: Customers can shop anytime, anywhere, and from any device according to their convenience.
 6. Product and price comparison: Customers can compare different products and prices from different shops on one website and make informed decisions.
 7. No reach limitations: Shop owners can expand their market and sell their products to customers from different regions, states, or even countries.
 8. Faster response to buyer/market demands: Shop owners can track customer behaviour, preferences, feedback, and trends using analytics tools and adjust their products accordingly.
 9. Centralized inventory management: Shop owners can keep stock in a single location and restock their store and fulfil orders from that central hub. This reduces inventory costs, errors, and wastage.

h. How do you generate revenue using the shopping cart?

- There are many ways to generate revenue using the shopping cart in such a system. Some of them are:
1. Offer one-click checkout: This can reduce the time and complexity of the checkout process and encourage customers to complete their purchase. We can use features such as saved payment methods, autofill forms, etc. to make it easier for customers to buy our products.
 2. Optimize the mobile shopping experience: This can ensure that our platform is responsive and user-friendly on mobile devices. We can use features such as mobile-friendly design, fast loading speed, easy navigation, etc. to make it convenient for customers to shop on their phones or tablets.
 3. Offer and promote “buy now, pay later” as payment options: This can increase customer confidence and flexibility in buying our products. We can use features such as instalment plans, deferred payments, etc. to allow customers to pay over time without interest or fees.
 4. Offer subscription and recurring payment services: This can increase customer loyalty and retention by providing them with regular deliveries of our products. We can use features such as automatic billing, customizable plans, etc. to offer customers a convenient and personalized service.
 5. Offer upsells and add-ons: This can increase our average order value by suggesting additional or complementary products that customers may be interested in buying. We can use features such as product bundles, cross-sells, etc. to show customers relevant offers based on their shopping cart contents.

3. Suppose, you are assigned to build Tree Plantation as a Service, named TPaaS.com, where the users buy trees to plant in particular selected locations virtually. On TPaaS.com, the selected tree is planted in a particular location by the staff of TPaaS.com company, and they look after the growth of the tree.

a. What are the key Cloud Computing components of TPaaS.com?

→ Some of the key cloud computing components of TPaaS are:

1. Client infrastructure: A front end platform that provides a user interface for users to buy trees and select locations. This can be a website or a mobile app that communicates with the back end platforms via APIs.
2. Back-end platforms: A back end platform that handles the business logic and data processing of the service. This can include servers that run applications such as inventory management, payment processing, order fulfilment, customer service etc. It can also include storage systems that store data such as user profiles, tree information, location details etc.
3. Application: A cloud based delivery model that allows users to access the service on demand without installing any software or hardware. This can be achieved by using a cloud service provider such as Microsoft Azure, Amazon Web Services (AWS), Google Cloud Platform (GCP) etc. These providers offer various services such as infrastructure as a service (IaaS), platform as a service (PaaS), software as a service (SaaS) etc. that can help reduce costs and increase scalability and reliability of the service.
4. Network: A network that connects the front end and back end platforms and enables data transfer between them. This can be either public or private depending on the security and performance requirements of the service.
5. Service: This is the component that connects the application with the back-end platforms and handles requests from users. It can be a web service, a cloud service, or a microservice.

b. How do you set up the company?

→ Some established steps that we can follow for setting up a cloud computing company with tree plantation as a service are:

1. Look for a business partner: We can collaborate with an existing reforestation organization or project that can help us with planting and maintaining the trees on the ground. We can choose from various options such as Digital Humani, Nelda Foundation or Sankalptaru.
2. Design some great marketing strategies: We need to promote our service and attract customers who are interested in planting trees online. We can use social media, blogs, podcasts, webinars, newsletters, or other channels to spread awareness and showcase our impact.

3. Buy required equipment: We need to invest in hardware and software that can support our cloud computing service. We may need servers, storage devices, routers, switches, firewalls, operating systems, databases, web frameworks, APIs, etc.
4. Arrange an office space: We need to find a suitable location to set up our office where we can manage our operations and staff. We may also need to hire employees such as developers, designers, marketers, accountants, etc.
5. Work according to the business plan: We need to follow our business plan and execute our strategies according to our goals and budget. We also need to monitor our performance and feedback from customers and partners.

c. What are the unique features of you would like to add to TPaaS.com?

→ Some of the unique features that can enhance TPaaS.com are:

1. Offering a variety of tree species and locations: We can allow our customers to choose from different types of trees and planting sites that suit their preferences and goals. We can also provide information about the benefits and challenges of each option.
2. Providing transparency and accountability: We can use technology such as GPS, satellite imagery, drones, blockchain, etc. to track and verify the planting and growth of the trees. We can also share regular updates and reports with our customers and partners about the impact of our service.
3. Engaging with local communities and stakeholders: We can involve and empower the local people who live near or depend on the forests. We can provide them with training, employment, education, health care, etc. We can also collaborate with other NGOs, governments, corporations, etc. who share our vision.
4. Creating a user-friendly and attractive interface: We can design a website or an app that is easy to use, secure, fast, and appealing for our customers. We can also use gamification, social media integration, rewards system, etc. to motivate and retain our customers.

d. How do you monetise the company?

→ There are different ways to monetize the company, such as:

1. Charging a fee per tree planted: We can set a price for each tree that our customers plant through our service. We can also offer discounts or incentives for bulk orders or subscriptions.
2. Selling carbon credits: We can generate and sell carbon credits from the trees that we plant and maintain. Carbon credits are certificates that represent a reduction of greenhouse gas emissions. We can sell them to individuals or organizations who want to offset their carbon footprint.

3. Offering additional services: We can provide other services related to tree planting, such as tree pruning, tree health management, stump removal, etc. We can also offer consultation or education services for customers who want to learn more about reforestation.
4. Creating partnerships and sponsorships: We can partner with other businesses or organizations who share our vision and values. We can also seek sponsorship from companies who want to support our cause or advertise their brand on our platform.

e. What are the legal issues (Law)? How do you overcome the legal issues?

→ Some of the legal issues that may arise in TPaaS company are:

1. Liability for tree damage or injury: We may be held liable if our trees cause damage or injury to someone else's property or person. We may have to pay for the repair, replacement, or compensation of the affected party.
2. Compliance with tree preservation laws: We may have to follow the laws and regulations of different states or countries regarding the protection and conservation of trees. We may need to obtain permission from the authorities before planting, cutting, or transplanting any trees.
3. Land rights and ownership disputes: We may face challenges in acquiring or accessing land for our tree planting activities. We may have to deal with conflicts with local communities, landowners, governments, or other stakeholders who have claims over the land.

To overcome these legal issues, you can:

1. Ensure proper insurance and risk management: We can obtain adequate insurance coverage for our trees and our operations. We can also implement risk management strategies such as regular inspection, maintenance, pruning, etc. of our trees.
2. Research and follow the relevant laws and regulations: We can conduct thorough research on the laws and regulations of each state or country where we operate. We can also consult with legal experts or authorities before undertaking any tree planting activities.
3. Engage and collaborate with local communities and stakeholders: We can involve and empower the local people who live near or depend on the forests. We can also seek their consent and cooperation before planting any trees on their land. We can also partner with other organizations who share our vision and values.

f. How do you find the locations to plant the trees?

→ There are different methods to find locations for tree planting, such as:

1. Collaborate with local communities and stakeholders: We may find the land and area to plant the trees by collaborating with local communities, farmers, governments or other stakeholders who have access to suitable land for tree plantation.

2. Using a planting zones map: You can use a map that shows the USDA gardening zones by ZIP code. This can help you find out which areas have suitable climate and soil conditions for different types of trees.
3. Inspecting the yard or site: We can check our yard or site for factors such as sunlight, water, nutrients, space, and drainage. These factors can affect the growth and health of the trees. We can also avoid planting near power lines, buildings, or other obstacles.
4. Following a planting guide: We can follow a step-by-step guide that shows us how to plant a tree properly. This can include tips on how to dig a hole, position the roots, fill the soil, water the tree, and mulch the area.

g. How do you attract clients?

- There are different ways to attract clients for a Tree Plantation as a Service company, depending on the target market, budget and goals. Some possible strategies are:
1. Use social media platforms like Facebook to grow and leverage word of mouth: We can create a page for our company, share photos and videos of our tree planting projects, invite reviews and testimonials from satisfied customers, and run ads or contests to reach more potential clients.
 2. Set up and optimize the Google My Business profile the right way: This is a free tool that helps us appear in local search results, maps and reviews when people look for tree planting services near them. We can add relevant information about our company, such as contact details, hours of operation, service areas and special offers.
 3. Offer a referral fee or incentive to existing customers who recommend our service to their friends, family or colleagues: This can help us generate more word of mouth and loyalty among our clients.
 4. Offer and market less competitive services that complement the core service of tree planting: For example, we can offer tree pruning, mulching, fertilizing or pest control services that can help maintain the health and beauty of the trees we plant.
 5. Partner with a non-profit organisation that shares the common vision of environmental conservation and social welfare: We can donate a portion of our profits to their cause, sponsor their events or campaigns, or collaborate on joint projects that can benefit both parties.
 6. Provide a certificate or recognition to the clients for their green initiative: This can help them feel good about their contribution to the environment and also showcase their social responsibility to others.

h. Why does a client believe in your services?

- A client should and does believe in our services because we provide:
1. Quality and professionalism: We use the best practices and standards for tree planting and care. We have a team of trained and experienced staff who can handle any project with efficiency and expertise. We also use high-quality saplings and materials that ensure the survival and growth of the trees we plant.
 2. Transparency and accountability: We keep our clients informed and updated about every step of the process through a public profile on the social media and/or the company website.
 3. Convenience and satisfaction: The client can plant trees online or offline without having to worry about the logistics, costs or maintenance of the project. The client can choose from a variety of locations and species that suit their preferences and goals. The client can also track the progress and impact of their tree planting through our website or app.
 4. Environmental and social benefits: The client can contribute to the restoration and conservation of natural habitats, biodiversity and ecosystem services. The client can also support local communities, farmers and organisations that benefit from the tree planting projects. The client can also reduce their carbon footprint and enhance their green image by planting trees.

4. Suppose, you are assigned to build a Secret Store as a Service, named SecretStore.com, where a client can store their data without worrying about the leakage. SecretStore guarantees that no one can steal the data from its database and the company people cannot retrieve and decrypt the data from their own database.

a. How do you achieve SecretStore.com? Elaborate.

→ To build your own Secret Store as a Service, you might want to consider some of the following steps:

1. Choose a cloud platform that offers key vault services: Example: Azure Key Vault or AWS Secrets Manager.
2. Create a key vault and generate or import secrets, such as passwords, connection strings, API keys, etc.: We can use the portal or the command-line interface of our cloud provider to do this.
3. Configure access policies and permissions for the key vault and secrets: We can use role-based access control (RBAC) or identity-based authentication to grant or deny access to your secrets.
4. Integrate the key vault with the applications using the Secrets Store CSI Driver: This will allow us to mount secrets as volumes in our pods. We can also use environment variables or SDKs to access our secrets programmatically.
5. Monitor and audit the key vault activity using logs and metrics: We can use tools like Azure Monitor or AWS CloudTrail to track who accessed our secrets and when.

b. How does a client benefit from the service?

→ A client can benefit from a Secret Store as a Service in several ways, such as:

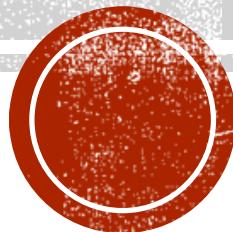
1. Enhanced security: A Secret Store as a Service can help protect the client's sensitive data from unauthorized access, leakage, or theft. The service can encrypt the secrets before storing them and use strong access policies and authentication mechanisms to prevent unauthorized access.
2. Reduced costs: A Secret Store as a Service can help reduce the costs of installing and maintaining physical infrastructure for storing secrets. The client can pay only for the storage space they need and scale up or down as needed.
3. Improved efficiency: A Secret Store as a Service can help improve the efficiency of the client's applications and workflows by providing easy and fast access to secrets. The service can also automate tasks such as secret rotation, revocation, and auditing.

c. What are the issues and challenges of the service?

- A Secret Store as a Service can also face some issues and challenges, such as:
1. Lack of visibility: It can be hard to keep track of all the secrets that are stored and used across different systems, resources, accounts, and applications. This can lead to secret sprawl, duplication, or leakage.
 2. Hardcoded credentials: Some secrets might be hardcoded or embedded within applications or systems, making them vulnerable to exposure or theft. Hardcoded credentials also make it difficult to rotate or revoke secrets when needed.
 3. Cloud computing privileges: Cloud environments often require more privileges and permissions for accessing secrets than on-premises environments. This can increase the risk of unauthorized access or misuse of secrets by malicious actors.
 4. Manual processes: Managing secrets manually can be time-consuming, error-prone, and inconsistent. Manual processes can also introduce human errors or delays that can compromise the security or availability of secrets.
 5. Lack of centralized management: Having a centralized system for managing secrets can help improve security, efficiency, and compliance. A centralized system can also provide features such as encryption, rotation, revocation, auditing, and backup of secrets.

SCHEDULING

Dr. Ripon Patgiri
Assistant Professor
National Institute of Technology Silchar



DELAY SCHEDULING [1]

Algorithm 2 Fair Sharing with Simple Delay Scheduling

```
Initialize  $j.skipcount$  to 0 for all jobs  $j$ .  
when a heartbeat is received from node  $n$ :  
    if  $n$  has a free slot then  
        sort  $jobs$  in increasing order of number of running tasks  
        for  $j$  in  $jobs$  do  
            if  $j$  has unlaunched task  $t$  with data on  $n$  then  
                launch  $t$  on  $n$   
                set  $j.skipcount = 0$   
            else if  $j$  has unlaunched task  $t$  then  
                if  $j.skipcount \geq D$  then  
                    launch  $t$  on  $n$   
                else  
                    set  $j.skipcount = j.skipcount + 1$   
                end if  
            end if  
        end for  
    end if
```



LATE SCHEDULING [2]

- Straggler task is a slow task that lags in submitting the work while other tasks complete the assigned work.
- LATE estimates the progress rate of each task as $\text{ProgressRate} = \frac{\text{ProgressScore}}{T}$, where T is the amount of time the task has been running for, and then estimate the time to completion as $\text{TimeToComplete} = \frac{1-\text{ProgressScore}}{\text{ProgressRate}}$
- LATE is based on three principles: prioritizing tasks to speculate, selecting fast nodes to run on, and capping speculative tasks to prevent thrashing.



REFERENCE

- [1] Zaharia, M., Borthakur, D., Sen Sarma, J., Elmeleegy, K., Shenker, S., & Stoica, I. (2010, April). Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling. In *Proceedings of the 5th European conference on Computer systems* (pp. 265-278).
- [2] Zaharia, Matei, Andy Konwinski, Anthony D. Joseph, Randy H. Katz, and Ion Stoica. "Improving MapReduce performance in heterogeneous environments." In *Osdi*, vol. 8, no. 4, p. 7. 2008.



Content I



Introduction

Syllabus & Text Book

Road map

Overview

Terminology

Taxonomy

Characteristics

On-demand self-service

Broad network access.

Measured service

Resource pooling

Rapid Elasticity

Service Models

Software as a Service

Platform as a Service

Infrastructure as a Service

Deployment Model

Content II



Private Cloud
Community Cloud
Public Cloud
Hybrid Cloud

Introduction

Syllabus & Text Book



CS 1482

Cloud Computing

3-0-0-6

8th sem (Open Elective)

Prerequisites: None

Introduction: Definition, Characteristics, Components, Cloud provider, SAAS, PAAS, IAAS and Others, Organizational scenarios of clouds, Administering & Monitoring cloud services, benefits and limitations, Deploy application over cloud, Comparison among SAAS, PAAS, IAAS

Cloud computing platforms: Infrastructure as service: Amazon EC2, Platform as Service: Google App Engine, Microsoft Azure, Utility Computing, Elastic Computing

Cloud Technologies: Study of Hypervisors, Compare SOAP and REST

Web services: SOAP and REST, SOAP versus REST, AJAX - asynchronous 'rich' interfaces, Mashups - user interface services

Virtualization: Virtual machine technology, virtualization applications in enterprises, Pitfalls of virtualization

Multitenant software: Multi-entity support, Multi-schema approach, Multi-tenancy using cloud data stores, Data access control for enterprise applications

Data in the cloud: Relational databases, Cloud file systems - GFS and HDFS, BigTable, HBase and Dynamo

Map-Reduce and extensions: Parallel computing, The map-Reduce model, Parallel efficiency of Map-Reduce, Relational operations using Map-Reduce, Enterprise batch processing using Map-Reduce, Introduction to cloud development, Example/Application of Mapreduce, Features and comparisons among GFS,HDFS etc, Map-Reduce model

Cloud security: Vulnerability assessment tool for cloud, Privacy and Security in cloud, Architectural Considerations - General Issues, Trusted Cloud computing, Secure Execution Environments and Communications, Security challenges - Virtualization security management- virtual threats, VM Security Recommendations, VM-Specific Security techniques, Secure Execution Environments and Communications in cloud

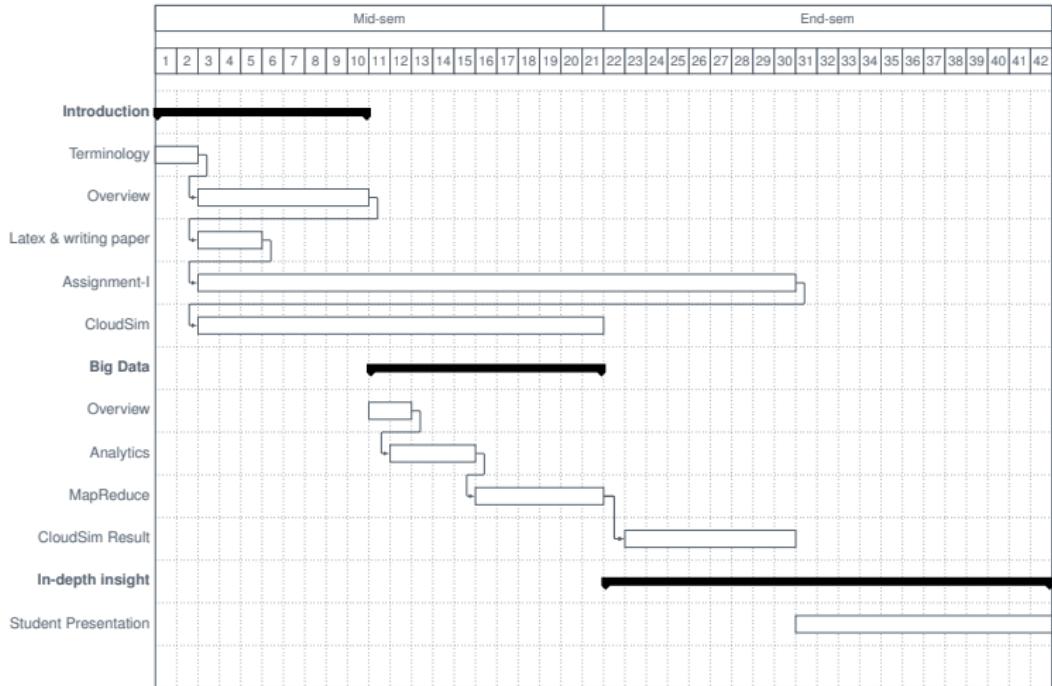
Issues: Implementing real time application over cloud platform
Issues in Intercloud environments, QOS Issues in Cloud, Dependability, data migration, streaming in Cloud. Quality of Service (QoS) monitoring in a Cloud computing environment

Books:

1. Cloud Computing for Dummies – Hurwitz J., Bloor R., Kanfman M., Halper F. (Wiley India)
2. Enterprise Cloud Computing – Shroff G. (Cambridge University Press)
3. Cloud Security – Krutz R., Vines R. D. (Wiley India)

Introduction

Road map





Definition

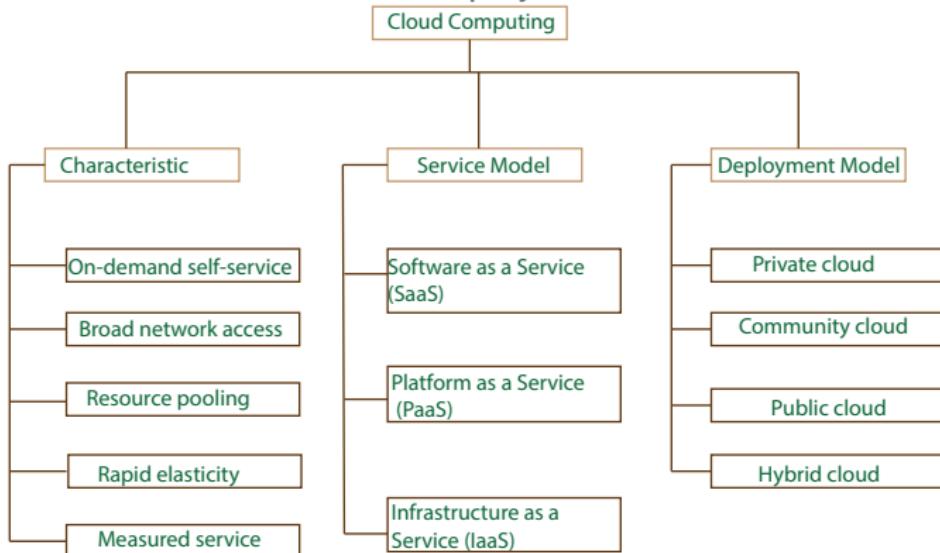
Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider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

Overview

Taxonomy



The cloud model is composed of five essential characteristics, three service models, and four deployment models.



Characteristics

On-demand self-service.



- ▶ A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider.

Characteristics

Broad network access.



- ▶ A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider.

Characteristics

Measured service



- ▶ Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts).
- ▶ Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service.

Characteristics

Resource pooling



10

This cloud model is composed of five essential characteristics, three service models, and four deployment models.

Characteristics

Resource pooling



10

This cloud model is composed of five essential characteristics, three service models, and four deployment models.

- ▶ The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand.

Characteristics

Resource pooling



10

This cloud model is composed of five essential characteristics, three service models, and four deployment models.

- ▶ The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand.
- ▶ There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or datacenter).
- ▶ Examples of resources include storage, processing, memory, and network bandwidth.

Characteristics

Rapid Elasticity



This cloud model is composed of five essential characteristics, three service models, and four deployment models.

- ▶ Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand.
- ▶ To the consumer, the capabilities available for provisioning often appear to be unlimited and can be appropriated in any quantity at any time.

Service Models

Software as a Service



12

- ▶ The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure.
- ▶ The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface.

Service Models

Software as a Service



- ▶ The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure.
- ▶ The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface.
- ▶ The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user specific application configuration settings.

Service Models

Platform as a Service



13

- ▶ The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider.

Service Models

Platform as a Service



- ▶ The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider.
- ▶ The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment.



- ▶ The capability provided to the consumer is to provision 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.



- ▶ The capability provided to the consumer is to provision 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.
- ▶ The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., host firewalls).

Deployment Model

Private Cloud



- ▶ The cloud infrastructure is provisioned for exclusive use by a single organization comprising multiple consumers (e.g., business units).
- ▶ It may be owned, managed, and operated by the organization, a third party, or some combination of them, and it may exist on or off premises.

Deployment Model

Community Cloud



16

- ▶ The cloud infrastructure is provisioned for exclusive use by a specific community of consumers from organizations that have shared concerns (e.g., mission, security requirements, policy, and compliance considerations).
- ▶ It may be owned, managed, and operated by one or more of the organizations in the community, a third party, or some combination of them, and it may exist **on or off premises**.

Deployment Model

Public Cloud



- ▶ The cloud infrastructure is provisioned for open use by the general public.
- ▶ It may be owned, managed, and operated by a business, academic, or government organization, or some combination of them.
- ▶ It exists on the premises of the cloud provider.

Deployment Model

Hybrid Cloud



- ▶ The cloud infrastructure is a composition of two or more distinct cloud infrastructures (private, community, or public) that remain unique entities, but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds).

Outline

1 Introduction to Cloud Computing

- Benefits of Cloud computing

2 Challenges of Cloud computing

3 Issues of Cloud computing

4 Obstacles of Cloud Computing

5 Organization Goals for Primary Storage

6 Organization Goals for Primary Storage

Benefits of Cloud Computing

Benefits of Cloud Computing

Cost Reduce Cost. **Pay-as-you-go** model reduce the cost of clients. Moreover, cloud computing reduces the initial establishment of business.

Storage Increases Storage. The user need not to worry about their storage size. Users are given elastic storage spaces.

Benefits of Cloud Computing

Benefits of Cloud Computing

Cost Reduce Cost. **Pay-as-you-go** model reduce the cost of clients. Moreover, cloud computing reduces the initial establishment of business.

Storage Increases Storage. The user need not to worry about their storage size. Users are given elastic storage spaces.

Flexibility Better. The manageability of business, software, or anything in the cloud is much easier than traditional computing model. Due to cloud provider, the business become more flexible.

Disaster An user need not to worry about their data. The cloud provider always ensure data recovery in disaster.

Maintenance User need not to bear the cost of maintenance and hence, business become much easier.

Challenges of Cloud Computing

Challenges of Cloud Computing

Data Transfer Performance and data transfer rates become key issues as the distance between the data and the user increases.

Latency Even unlimited bandwidth without solving the latency problem will not improve the performance because it is the latency.

Challenges of Cloud Computing

Challenges of Cloud Computing

- Data Transfer** Performance and data transfer rates become key issues as the distance between the data and the user increases.
- Latency** Even unlimited bandwidth without solving the latency problem will not improve the performance because it is the latency.
- Bandwidth** Not all data access patterns are well suited to the cloud, particularly if there are large distances to cover. In such cases, bandwidth becomes not only a challenge but a financial consideration.
- Requirements** User requirements are unlimited bandwidth, high performance, low-response time, and mission-critical data.
- Storage** Maintenance of storage.

Issues of Cloud Computing

Issues of Cloud Computing

Security Security & Data Availability

Costs Costs.

Performance Performance

Latency Latency constraints

Management Manageability

Issues of Cloud Computing

Issues of Cloud Computing

Security Security & Data Availability

Costs Costs.

Performance Performance

Latency Latency constraints

Management Manageability

Interoperability A serious concern exists today is: Most of todays on-premises applications use block protocols such as FC, iSCSI etc. But Cloud storage protocols predominantly speak only in the language of file protocols (CIFS, NFS) and both public and private storage clouds are accessed via REST [] HTTP-based, or SOAP APIs. the figure ??, says that <http://cse.nits.ac.in/prasenjit/>

Contd.

Obstacles of Cloud Computing

There is always other side of a coin

Cloud Computing Adoption Obstacles

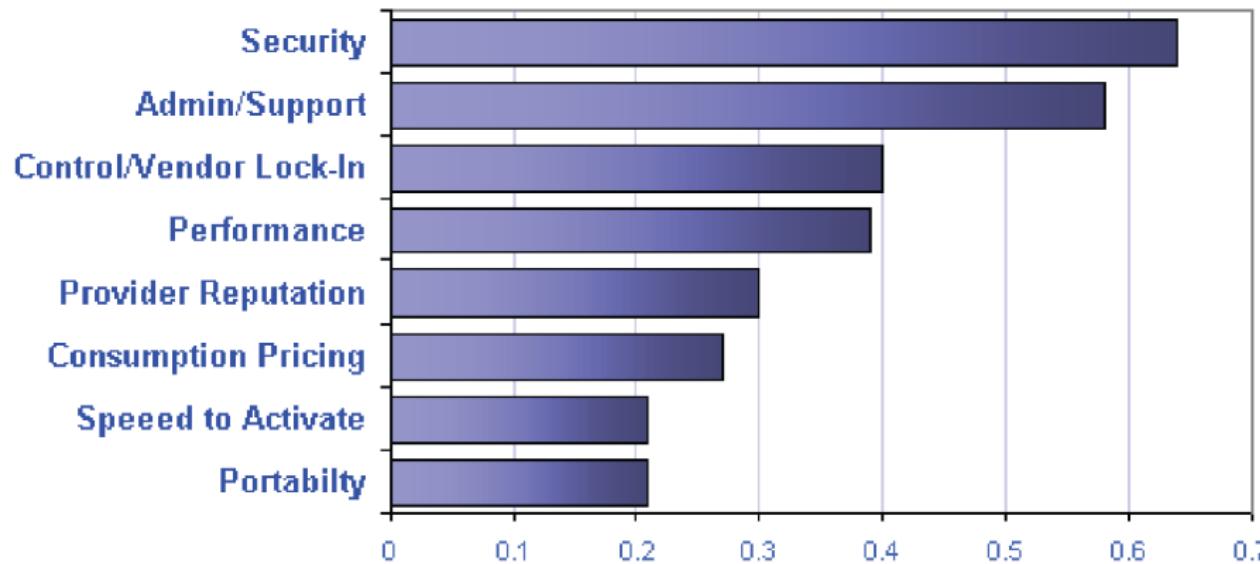


Figure: Obstacle of Cloud Computing

Organization Goals for Primary Storage

Organization Goals for Primary Storage

Economic Improved storage economics Leverage appropriate tiers of storage according to the performance requirement of the data.

Performance consistency working set and hotspot data to be automatically tiered to the highest-performance tier of storage (SSD), whereas non working-set or non hotspot data is tiered to a lower performing tier dynamically

Organization Goals for Primary Storage

Organization Goals for Primary Storage

Economic Improved storage economics Leverage appropriate tiers of storage according to the performance requirement of the data.

Performance consistency working set and hotspot data to be automatically tiered to the highest-performance tier of storage (SSD), whereas non working-set or non hotspot data is tiered to a lower performing tier dynamically

Transparent integration no complex policies to manage, no agents to install on servers, and the storage view from server to remain same in light of new integration with cloud storage

Deduplication - of primary storage to eliminate the repeated storage of redundant segments of data.

Organization Goals for Primary Storage

Organization Goals for Primary Storage

Economic Improved storage economics Leverage appropriate tiers of storage according to the performance requirement of the data.

Performance consistency working set and hotspot data to be automatically tiered to the highest-performance tier of storage (SSD), whereas non working-set or non hotspot data is tiered to a lower performing tier dynamically

Transparent integration no complex policies to manage, no agents to install on servers, and the storage view from server to remain same in light of new integration with cloud storage

Deduplication - of primary storage to eliminate the repeated storage of redundant segments of data.

Compression - Achieve higher levels of compression even when encountering single-byte insertion scenarios.

Data Reduction - Achieve 10X data reduction for specific storage workloads.

Organization Goals for Primary Storage

Organization Goals for Primary Storage

Version control support Version control can be enabled, thanks to some companies' implementation of primary storage de-duplication, which minimizes storage capacity requirements and cost

Pay-as-you-grow use of cloud storage enables pay-as-you-grow capacity consumption, which minimizes cost of over-provisioned yet unutilized on-premises storage

Organization Goals for Primary Storage

Organization Goals for Primary Storage

Version control support Version control can be enabled, thanks to some companies' implementation of primary storage de-duplication, which minimizes storage capacity requirements and cost

Pay-as-you-grow use of cloud storage enables pay-as-you-grow capacity consumption, which minimizes cost of over-provisioned yet unutilized on-premises storage

Optimal Cost Structure Using de-duplication, compression and encryption minimize cost for cloud storage capacity and cost for IO and data transfer,

Tape elimination Use Cloud Clones to enable consistent, point-in-time recovery snapshots and store independent copies in the cloud to eliminate need for tape

Outline

- 1 About the Paper
- 2 Provisioning
 - Discover
 - Selection
 - Allocation
- 3 Portability
 - VM Mobility
 - Data Portability
- 4 Security
- 5 Service-Level Agreement (SLA)
 - Trust
 - Authorization & Identity management
 - Federated SLA Management
 - Federated-Level Agreement
- 6 SLA Dependency
- 7 Legal Issues
- 8 Monitoring
- 9 Network
 - Connectivity
 - Addressing
 - Naming
 - Multicasting
- 10 Autonomy

Reference paper

Reference

Title: Interconnected Cloud Computing Environments:
Challenges, Taxonomy, and Survey

Author: ADEL NADJARAN TOOSI, RODRIGO N. CALHEIROS,
and RAJKUMAR BUYYA.

Affiliation: The University of Melbourne, Australia.

Journal: ACM Computing Survey, Volume 47, Issue 1, Article 7
(April 2014), 47 pages.

DOI: <http://dx.doi.org/10.1145/2593512>

Provisioning

Discover

- Cloud service discovery allows automatic detection of services and resources offered by cloud providers on the Internet. Since cloud providers offer a variety of services and use different ways to describe them, a way to provide a common access to cloud services and to discover and deploy them is necessary.

Provisioning

Discover

- Cloud service discovery allows automatic detection of services and resources offered by cloud providers on the Internet. Since cloud providers offer a variety of services and use different ways to describe them, a way to provide a common access to cloud services and to discover and deploy them is necessary.

Provisioning

Discover

- Cloud service discovery allows automatic detection of services and resources offered by cloud providers on the Internet. Since cloud providers offer a variety of services and use different ways to describe them, a way to provide a common access to cloud services and to discover and deploy them is necessary.
- Cloud customers require selection of the best possible application deployments in the cloud according to their objectives and constraints for QoS.

Provisioning

Discover

- Cloud service discovery allows automatic detection of services and resources offered by cloud providers on the Internet. Since cloud providers offer a variety of services and use different ways to describe them, a way to provide a common access to cloud services and to discover and deploy them is necessary.
- Cloud customers require selection of the best possible application deployments in the cloud according to their objectives and constraints for QoS.
- For instance, Google App Engine and Amazon EC2 do not offer discovery services, and Microsoft Azure and Force.com offer limited discovery.

Provisioning

Discover

- One of the main issues regarding service discovery in a multiple-cloud deployment is the lack of an integrated repository of cloud services.

Provisioning

Discover

- One of the main issues regarding service discovery in a multiple-cloud deployment is the lack of an integrated repository of cloud services.
- Another issue is that cloud providers describe their services with diverse languages, terms, and names. Moreover, there is not a common understanding regarding service functionalities, their QoS, and metrics among providers and customers.

Provisioning

Discover

- One of the main issues regarding service discovery in a multiple-cloud deployment is the lack of an integrated repository of cloud services.
- Another issue is that cloud providers describe their services with diverse languages, terms, and names. Moreover, there is not a common understanding regarding service functionalities, their QoS, and metrics among providers and customers.
- Finally, states of a large part of services in clouds change constantly and are dynamic in nature. The situation is even worse in interconnected cloud environments.

Provisioning

Selection

- Optimal application deployment in the cloud requires an effective selection strategy that works based on QoS criteria such as reliability, cost, and security and returns the set of the most suitable cloud services for end-customers.

Provisioning

Selection

- Optimal application deployment in the cloud requires an effective selection strategy that works based on QoS criteria such as reliability, cost, and security and returns the set of the most suitable cloud services for end-customers.
- Currently, selection is performed manually by cloud customers based on their requirements or through consultant companies.

Provisioning

Selection

- Optimal application deployment in the cloud requires an effective selection strategy that works based on QoS criteria such as reliability, cost, and security and returns the set of the most suitable cloud services for end-customers.
- Currently, selection is performed manually by cloud customers based on their requirements or through consultant companies.
- However, application deployment across multiple providers benefits from significant features such as the range of geographical locations, lower latency, higher reliability, lower deployment cost, higher failure resistance, and so forth.

Provisioning

Selection

- Optimal application deployment in the cloud requires an effective selection strategy that works based on QoS criteria such as reliability, cost, and security and returns the set of the most suitable cloud services for end-customers.
- Currently, selection is performed manually by cloud customers based on their requirements or through consultant companies.
- However, application deployment across multiple providers benefits from significant features such as the range of geographical locations, lower latency, higher reliability, lower deployment cost, higher failure resistance, and so forth.
- Consequently, an automated selection approach for application deployment is well motivated to optimize different aspects such as latency, reliability, throughput, data transfer, and cost.

Provisioning

Allocation

- Cloud providers usually offer their virtualized resources based on different QoS levels (e.g., best effort and reserved).

Provisioning

Allocation

- Cloud providers usually offer their virtualized resources based on different QoS levels (e.g., best effort and reserved).
- Physical resources in clouds are shared between cloud users.
Therefore, allocation strategies are needed to allocate resources to the requests in a profitable manner while fulfilling requests QoS requirements.

Provisioning

Allocation

- Cloud providers usually offer their virtualized resources based on different QoS levels (e.g., best effort and reserved).
- Physical resources in clouds are shared between cloud users. Therefore, allocation strategies are needed to allocate resources to the requests in a profitable manner while fulfilling requests QoS requirements.
- Contention happens when a user request cannot be admitted or cannot acquire sufficient resources because resources are occupied by other requests. This phenomenon is called resource contention.

Portability

VM Mobility

- VM Mobility is defined as the ability to move a running VM from one host to another without stopping it.

Portability

VM Mobility

- VM Mobility is defined as the ability to move a running VM from one host to another without stopping it.
- VM mobility requires- Memory and state transfer between hosts residing in different data centers.

Portability

VM Mobility

- VM Mobility is defined as the ability to move a running VM from one host to another without stopping it.
- VM mobility requires- Memory and state transfer between hosts residing in different data centers.
- VM mobility requires- Same LAN access by VMs at the destination host, without two sites sharing LAN.

Portability

VM Mobility

- VM Mobility is defined as the ability to move a running VM from one host to another without stopping it.
- VM mobility requires- Memory and state transfer between hosts residing in different data centers.
- VM mobility requires- Same LAN access by VMs at the destination host, without two sites sharing LAN.
- VM mobility requires- Same storage access by VMs at the destination host, without two sites sharing storage.

Portability

VM Mobility

- VM Mobility is defined as the ability to move a running VM from one host to another without stopping it.
- VM mobility requires- Memory and state transfer between hosts residing in different data centers.
- VM mobility requires- Same LAN access by VMs at the destination host, without two sites sharing LAN.
- VM mobility requires- Same storage access by VMs at the destination host, without two sites sharing storage.
- VM Mobility requires- a VM transfer from one physical machine to another without disrupting the network traffic flow.

Data Portability

Data Portability

- A cloud provider stores data in their own proprietary format, then users cannot move their data to other vendors without considerable cost and technical effort.

Data Portability

Data Portability

- A cloud provider stores data in their own proprietary format, then users cannot move their data to other vendors without considerable cost and technical effort.
- Therefore, industry standards and exporting tools, or at the very least formats that are publicly documented, are required to avoid data lock-in.

Data Portability

Data Portability

- A cloud provider stores data in their own proprietary format, then users cannot move their data to other vendors without considerable cost and technical effort.
- Therefore, industry standards and exporting tools, or at the very least formats that are publicly documented, are required to avoid data lock-in.
- Data portability is hindered by the lack of proper technology and standards and non-portability of the applications and data, which is exploited by cloud service providers for their own benefits

Data Portability

Avoid Vendor Lock-In

- Using APIs that have multiple independent implementations, for example, Amazon EC2 APIs, which are used by several others such as Eucalyptus.

Data Portability

Avoid Vendor Lock-In

- Using APIs that have multiple independent implementations, for example, Amazon EC2 APIs, which are used by several others such as Eucalyptus.
- Choosing a particular API that can run on multiple Clouds, for example, MapReduce and Hadoop.

Data Portability

Avoid Vendor Lock-In

- Using APIs that have multiple independent implementations, for example, Amazon EC2 APIs, which are used by several others such as Eucalyptus.
- Choosing a particular API that can run on multiple Clouds, for example, MapReduce and Hadoop.
- Manually decoupling the cloud-specific code of the application designed for each cloud provider from the application logic layer.

Data Portability

Avoid Vendor Lock-In

- Using APIs that have multiple independent implementations, for example, Amazon EC2 APIs, which are used by several others such as Eucalyptus.
- Choosing a particular API that can run on multiple Clouds, for example, MapReduce and Hadoop.
- Manually decoupling the cloud-specific code of the application designed for each cloud provider from the application logic layer.
- Creation of widespread standards and APIs.

Data Portability

Avoid Vendor Lock-In

- Using APIs that have multiple independent implementations, for example, Amazon EC2 APIs, which are used by several others such as Eucalyptus.
- Choosing a particular API that can run on multiple Clouds, for example, MapReduce and Hadoop.
- Manually decoupling the cloud-specific code of the application designed for each cloud provider from the application logic layer.
- Creation of widespread standards and APIs.
- Utilization of vendor-independent cloud abstraction layers such as jclouds and libcloud.

Security

Trust

- Trust typically refers to a situation where one party is willing to rely on the actions of another party.

- Trust typically refers to a situation where one party is willing to rely on the actions of another party.
- Ultimately, there is uncertainty as to whether the trusted party will behave or deliver as promised.

- Trust typically refers to a situation where one party is willing to rely on the actions of another party.
- Ultimately, there is uncertainty as to whether the trusted party will behave or deliver as promised.
- In the cloud computing environment, customers must trust in a cloud provider for the privacy and security of their assets (i.e., their data and processes).

- Trust typically refers to a situation where one party is willing to rely on the actions of another party.
- Ultimately, there is uncertainty as to whether the trusted party will behave or deliver as promised.
- In the cloud computing environment, customers must trust in a cloud provider for the privacy and security of their assets (i.e., their data and processes).
- In cloud computing, the risk of losing data confidentiality, integrity, and availability for customers is triggered by the lack of control over the data and processes.

- In the Inter-cloud scenario, the trust and reputation of a cloud provider affect other cloud providers.

- In the Inter-cloud scenario, the trust and reputation of a cloud provider affect other cloud providers.
- Trust federation is a combination of technology and policy infrastructure that allows organizations to trust each others verified users to enable the sharing of information, resources, and services in a secure and distributed way.

Security

Authorization & Identity management

- Identity Management (IdM) is an administrative task that deals with authentication of individuals in a system and authorization to access resources of the system based on the associated rights and restrictions.

Security

Authorization & Identity management

- Identity Management (IdM) is an administrative task that deals with authentication of individuals in a system and authorization to access resources of the system based on the associated rights and restrictions.
- In cloud computing environments, identity management services are mainly responsible for supporting access control to services based on user attributes (e.g., IP address, user and group name) and resource attributes (e.g., availability periods).

Security

Authorization & Identity management

- Identity Management (IdM) is an administrative task that deals with authentication of individuals in a system and authorization to access resources of the system based on the associated rights and restrictions.
- In cloud computing environments, identity management services are mainly responsible for supporting access control to services based on user attributes (e.g., IP address, user and group name) and resource attributes (e.g., availability periods).
- Identity management systems, in federated cloud environments, should allow identification of users and resources in addition to support interoperability across multiple identity domains.

Security

Authorization & Identity management

- Single Sign-On (SSO) authentication, where a cloud must be able to authenticate itself to gain access to the resources provided by federated foreign clouds belonging to the same trust context without further identity checks.

Security

Authorization & Identity management

- Single Sign-On (SSO) authentication, where a cloud must be able to authenticate itself to gain access to the resources provided by federated foreign clouds belonging to the same trust context without further identity checks.
- Digital identities and third parties, where a cloud has to be considered as a subject distinctively identified by credentials and each cloud must be able to authenticate itself with foreign clouds using its digital identity guaranteed by a third party.

- Cloud providers define (or negotiate with customers) a service-level agreement (SLA) to specify what they guarantee.

- Cloud providers define (or negotiate with customers) a service-level agreement (SLA) to specify what they guarantee.
- SLA is a contract that describes a service and, most importantly, sets the expected service-level objectives (QoS expectations).

- Cloud providers define (or negotiate with customers) a service-level agreement (SLA) to specify what they guarantee.
- SLA is a contract that describes a service and, most importantly, sets the expected service-level objectives (QoS expectations).
- Implementation of SLA mechanisms on top of federated resources is still an open question.

- In federated cloud environments, each participant cloud provider has its own SLA management mechanisms.

- In federated cloud environments, each participant cloud provider has its own SLA management mechanisms.
- The methods and protocols for negotiation of dynamic and flexible SLAs is required for dynamic environments such as Inter-cloud.

- In federated cloud environments, each participant cloud provider has its own SLA management mechanisms.
- The methods and protocols for negotiation of dynamic and flexible SLAs is required for dynamic environments such as Inter-cloud.
- An important issue in the federated cloud environment is how SLAs can be enforced in a federation where there are conflicting policies and goals of different members versus the objectives of the federation as a whole.

SLA

Federated-Level Agreement

- Federation-Level Agreement (FLA) that includes the set of rules and conditions that has to be signed by new providers once they join the federation.

- Federation-Level Agreement (FLA) that includes the set of rules and conditions that has to be signed by new providers once they join the federation.
- For example, a federation can set rules for minimum resources contributed to the federation or the set of QoS such as minimum expected availability.

- In federated cloud environments where a provider outsources its load to another provider, it expects a set of guaranteed QoS that is compatible to the promised QoS to end-users.

- In federated cloud environments where a provider outsources its load to another provider, it expects a set of guaranteed QoS that is compatible to the promised QoS to end-users.
- Quality of a service can be affected by external services.

- In federated cloud environments where a provider outsources its load to another provider, it expects a set of guaranteed QoS that is compatible to the promised QoS to end-users.
- Quality of a service can be affected by external services.
- It means that if one of the lower-layer services (e.g., infrastructure layer) is not functioning properly, it can affect the performance of higher-layer services (e.g., software layer).

- In federated cloud environments where a provider outsources its load to another provider, it expects a set of guaranteed QoS that is compatible to the promised QoS to end-users.
- Quality of a service can be affected by external services.
- It means that if one of the lower-layer services (e.g., infrastructure layer) is not functioning properly, it can affect the performance of higher-layer services (e.g., software layer).
- A practical approach is required to model the dependencies among services.

- Cloud computing by itself requires consideration of a broad set of legal issues such as privacy and security, contracting issues, issues related to location and ownership of data, and business considerations.

- Cloud computing by itself requires consideration of a broad set of legal issues such as privacy and security, contracting issues, issues related to location and ownership of data, and business considerations.
- When different organizations are involved in providing services for customers, one major issue is that it is difficult to guarantee confidentiality and privacy on data, especially when data is located in different countries with different laws.

- Cloud computing by itself requires consideration of a broad set of legal issues such as privacy and security, contracting issues, issues related to location and ownership of data, and business considerations.
- When different organizations are involved in providing services for customers, one major issue is that it is difficult to guarantee confidentiality and privacy on data, especially when data is located in different countries with different laws.
- For example, in a federated scenario in which a cloud provider leverages another providers services and customers might not generally have control of where the service they are leasing is operating, in case of failures in the service delivery it will be difficult for the cloud user to identify the real causes.

Cloud Computing - Quick Notes:

- Cloud refers to the 'servers' that are accessed over the internet, present at remote location.
- Cloud computing is the on-demand availability of computer system resources without direct active management by the user.
In short, storing, managing and process data on remote servers.
- Service Providers: (i) Google Cloud (ii) AWS (iii) MS Azure (iv) IBM Cloud.
- Types:
 - I) Public : Accessible for all
 - II) Private : Services accessible within an org.
 - III) Hybrid : public + private cloud features.
 - IV) Community : Services accessible by a group of orgs.
- Characteristics:
 1. On-demand self service: consumer can request and receive access to a service without third-party's (admin/staff) approval or need to accept request.
 2. Broad network Access: Access anywhere and anytime.
 3. Resource Pooling: Multiple customers, same physical resources.
 4. Measured Services: Pay according to services used.
 5. Rapid Elasticity and Scalability: Ability to quickly provision resources in the cloud as the org. need them.
 6. Easy or Zero Maintenance: Customer need not worry about resource failures.
 7. Security : Minimal data loss / failure as copy is stored on multiple servers and not just one. hence, data safe.

Advantages:

1. Resource accessible anywhere, anytime.
2. On-demand self service.
3. Reduced IT cost (^{Hardware Purchase}_{not required})
4. Scalability.
5. Collaboration over different region.
6. Security, No data loss / failure.
7. Location and device independence.
8. We need not update software.
9. Customer's maintenance not required hence, saves our time.

Disadvantages

1. Network Connection Dependency
2. Lack of Support if any resource is not working, hence trustworthy service provider is a must.
3. May not get all features | Limited accessibility | Customers' don't have say (not in control) (Provider-chosen features)
4. Vendor-lock-in problem.

* Vendor lock-in problems:

Situation where customers are dependent (i.e., locked in) on a single cloud provider technology, implementation and cannot easily move on in the future to a different vendor/service provider without substantial cost, legal constraint or technical incompatibilities i.e., org. can/may face problem when transferring their services from one vendor to another.

- Types of Vendor lock-in Risks:

1. Data Transfer Risk: format of data extracted, cryptography or security mechanism used to encrypt data.

2. Application Transfer Risk: Compatibility, service provider's partnership cost, reconfiguration of app non-natively is expensive and difficult, lack of standards.

3. Human Resource Knowledge Risk: Employee's knowledge of the new platform may not be as good as the old platform. Relearning time.

* Cloud Computing Architecture: (2 parts: frontend & backend).

- front-end :: used by client

- contains all client side interfaces and apps required to access the cloud platform.

- back-end :: used by service provider

- manages all resources required to provide C.C. services
- includes huge amt. of data storage, security mechanisms, virtual machines, deployment models etc



* Components of C.C. Architecture:

1. Client Infrastructure: frontend/GUI

2. Application: Maybe software or platform.

3. Service: SaaS, PaaS, IaaS

4. Runtime Cloud: Execution & Runtime Env. to C/M.

5. Storage.

6. Infrastructure: HW & S/W components, Server

7. Management.

8. Security

9. Internet

* SaaS (Software as a Service):

- Way of delivering services and apps. over the internet.
- Maintenance of SW & HW done by vendor.
- Removes cost of HW and SW maintenance.
- Used as a finished product by the ^{end} users, can't make changes by themselves. E.g.: ~~Word, Excel, Powerpoint~~, Gmail, G-slide, MS Team, Dropbox etc.

- Characteristics:

- SW available over internet
- SW maintained by vendor
- Cost effective
- Available on-demand.
- Scaled up/down as per need.
- Worked on Shared Model, i.e., one SW → multiple clients, using registration accounts.
- SW automatically upgraded.

- Benefits:

- Platform independent
- Multi-tenant Solutions
- Scale up / Scale down.
- Accessible anytime, anywhere.
- Reduced time (accessible directly from browser).
- Cost effective (pay as per use)

* PaaS (Platform as a Service):

- Developers use it
- Provides platform & env. (i.e., runtime env) to build apps. & services
- Offers development and deployment tools.
- Hosted in cloud & accessed by user via web browser.
- No control over the infrastructure. Only interact with the UI and OS provided by the vendor. No control over it.

Advantages :-

- Pay as per use, cost effective
- No need to purchase expensive servers, SW or storage.
- Scale up/down anytime.
- SW management (updates) done by provider.
- Easy deployment of web applications.

* IaaS (Infrastructure as a Service)

- Provides infrastructure.
- Used by system admin / or network architects (full control)
- Provides underlying O.S., security, networking and servers.
- Provides access to fundamental resources such as physical machine, virtual machine, storage etc.

Advantages:-

- Scale up / down as required.
 - Cost effective (pay as use)
 - full control over resources.
- Offers :
- Virtual machine disk storage
 - IP address
 - Virtual LAN (VLAN)
 - Load balancer.

Example: AWS, IBM Cloud, Azure.

IaaS	PaaS	SaaS
Application Data Runtime Middleware OS	Application Data	Nothing
Virtualization Servers Storage Networking	Routine Middleware OS Virtualization Server Storage Networking	Application Data Routine Middleware OS Virtualization Server Storage Networking

end-user
manages
or service
user manages

Service
provider
manages.

* Applications of C.C.

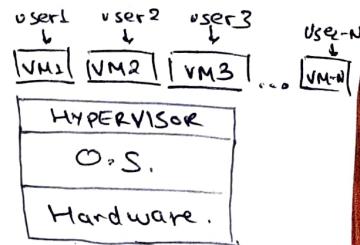
- 1) Business Application : E.g.: Salesforce, Paypal etc.
- 2) Data Storage & Backup Apps : E.g.: Google Drive, OneDrive, etc.
- 3) Educational Application : E.g.: Google Docs, Chromebook for edu, etc.
- 4) Entertainment App : E.g.: Online games, Video Conferencing, etc.
- 5) Art/Media App : E.g.: Photopea, Travid.live, etc.
- 6) Social Applications : E.g.: Twitter, FB, IG, etc.

* Types of Cloud :

- 1) Public :
 - Pay as per use (for services)
 - managed by third parties (provider)
 - fundamental characteristics : MULTITENANCY (Shared resource)
 - Advantages - maintenance, on-demand, anywhere, anytime, scalable
 - Disadvantages -
 - 1. less secure b/c resources publically shared
 - 2. less customizable as compared to private cloud.
- 2) Private :
 - Service accessible within an org.
 - called internal / corporate cloud
 - managed by either org. or 3rd party.
 - Advantages - high security, more customizable, improved reliability
 - Disadvantages - Limited operations, high cost, limited scalability
- 3) Hybrid :
 - Critical activities performed / handled by private cloud.
 - Non-critical activities performed / handled by public cloud.
 - Advantages - Scalability, Security, Low Cost, flexibility
 - Disadvantages - Maintenance, Management, Dependency on infrastructure
- 4) Community :
 - Services accessible by group of orgs' to share info b/w orgs
 - owned / managed by 1 or more orgs. in community or by 3rd party
 - Advantages - cost effective, shared Resource, Secure
 - Disadvantages - Data Safety as shared resource, Maintenance, Increased cost

* Virtualisation in Cloud Computing :

- Technique that allows to share single physical instance of an app or resource among multiple org. or customers.
 - Software called 'Hypervisor' deals with / manages virtualisation.
 - All virtual resources will work independently.
- Host Machine : Machine where virtual machine is actually built (on provider's side)
- Guest Machine : Virtual machine (on user's side).
- Hypervisor (Virtual Machine Monitor, VMM)
 - E.g.: VMware, Hyper-V
 - SW that creates and runs Virtual Machines.
 - Types:
 - Type 1 hypervisor (bare metal or native hypervisor)
 - Type 2 hypervisor (hosted or embedded hypervisor)



• Advantages:

- i) better resource utilisation
- ii) lowers the cost of IT infrastructure
- iii) Remote Access
- iv) Pay per use of IT infrastructure on demand.
- v) Enables running multiple O.S.
- vi) if one virtual machine is not working or having problem, others will not be affected.

* Serverless Computing:

- Cloud computing execution model in which cloud service provider allocates resources on demand, taking care of the servers on behalf of their customers.
 - no infrastructure management
 - autoscaling; based on incoming requests
- managed by cloud provider
- Serverless Architecture is a way to build and run applications and services without having to manage infrastructure.
- Basically, SaaS and PaaS are serverless computing services.
- Reduces cost / cost effective (no charge for idle time).
- When invoked, Runs for a short duration only; i.e., when the app is not in use, there are no computing resources allocated to that app.
- Application : E.g.: Weather update component in application, i.e., Gets invoked only when clicked, other time, idle or does not update automatically.