CLOUD COMPUTING ARCHITECTURE

BY

ANIMESH CHATURVEDI

https://sites.google.com/site/animeshchaturvedi07/

Cloud Computing Architecture

1. Requirements
2. Introduction Cloud computing architecture
3. Various kind of Cloud computing architecture
4. SOA, Grid and Cloud Computing
5. Transactional, On Demand & Distributed Computing

https://sites.google.com/site/animeshchaturvedi07/academic-teaching/cloudcomputing

Requirements

Cloud computing is required to meet the requirement of scalability, cost efficiency, legal agreement and business.

https://sites.google.com/site/animeshchaturvedi07/academic-teaching/cloudcomputing

Changing requirement and roles

Changes are required for IT and software architecture like

- Data (storage, distribution),
- High processing computation
- Transactions computing
- Caching
- Workflows
- Access control.
- Service Level Agreement requirements

Changing enterprise and industrial environments in the roles of customer, provider, and ISV

Architecture for Elasticity

- **Vertical Scale-Up**
  - Keep on adding resources to a unit to increase computation power.
  - Process the job to single computation unit with high resources.
- **Horizontal Scale Out**
  - Keep on adding discrete resources for computation and make them behave as in converged unit.
  - Splitting job on multiple discrete machines, combine the output.
  - Distribute database.
- For HPC second option is better than first. Because Complexity and cost of first option is very high.

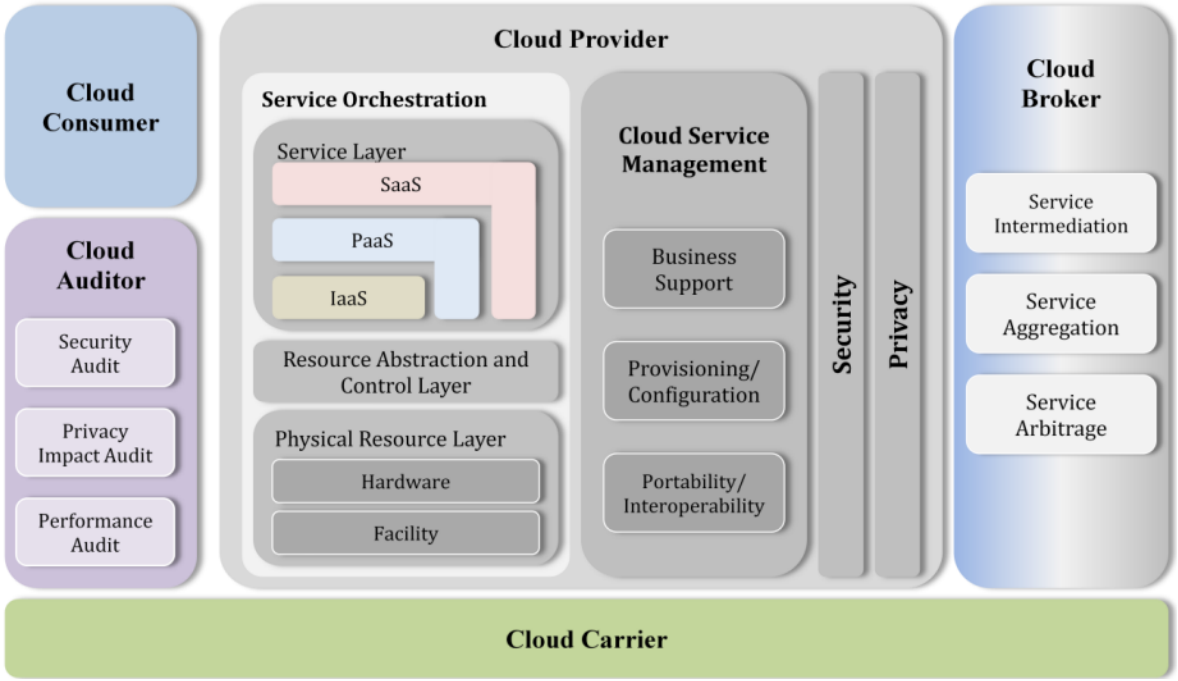
Introduction Cloud computing architecture

https://sites.google.com/site/animeshchaturvedi07/academic-teaching/cloudcomputing

Cloud Conceptual Reference Model

- Cloud High-level architecture
- Five major actors with their roles, responsibilities, activities and functions in cloud computing.
- Understanding of the requirements, uses, characteristics and standards of cloud computing.

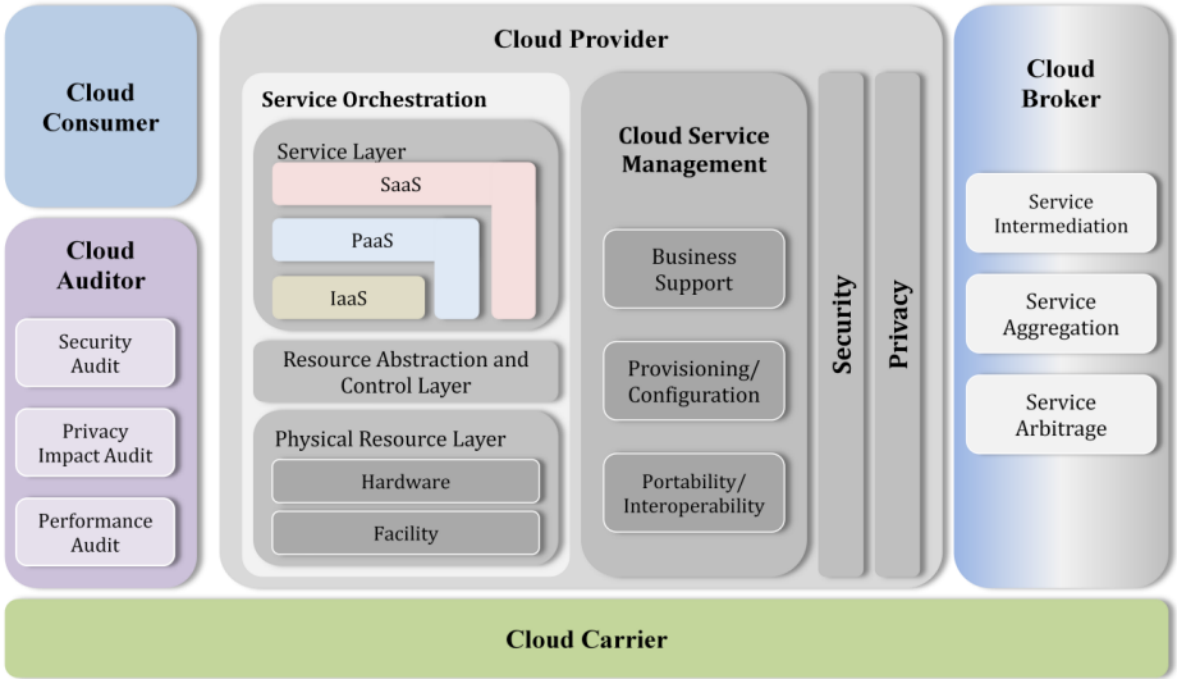
1. *Cloud Consumer*
2. *Cloud Provider*
3. *Cloud Broker*
4. *Cloud Auditor*
5. *Cloud Carrier*



NIST Cloud Computing Reference Architecture

Cloud Service Characteristics

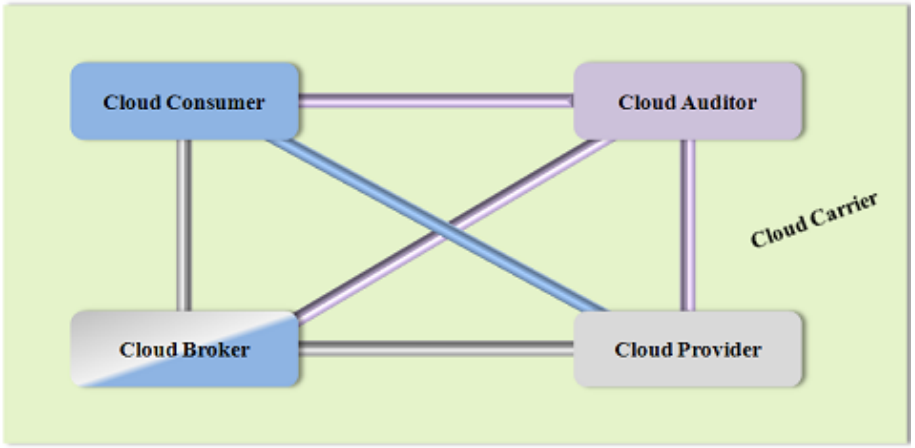
- *On demand self-service*
- *Broad network access*
- *Resource pooling*
- *Rapid elasticity*
- *Measured service*



NIST Cloud Computing Reference Architecture

Actors in Cloud Computing

- **Cloud Consumer** A person or organization that maintains a business relationship with, and uses service from, *Cloud Providers*.
- **Cloud Provider** A person, organization, or entity responsible for making a service available to interested parties.
- **Cloud Auditor** A party that can conduct independent assessment of cloud services, information system operations, performance and security of the cloud implementation.
- **Cloud Broker** An entity that manages the use, performance and delivery of cloud services, and negotiates relationships between *Cloud Providers* and *Cloud Consumers*.
- **Cloud Carrier** An intermediary that provides connectivity and transport of cloud services from *Cloud Providers* to *Cloud Consumers*.



NIST Cloud Computing Reference Architecture

Scenarios in Cloud: 1

1. Cloud consumer interacts with the cloud broker instead of contacting a cloud provider directly.
2. The cloud broker may create a new service (mash up) by combining multiple services or by enhancing an existing service.
3. Actual cloud providers are invisible to the cloud consumer.

NIST Cloud Computing Reference Architecture

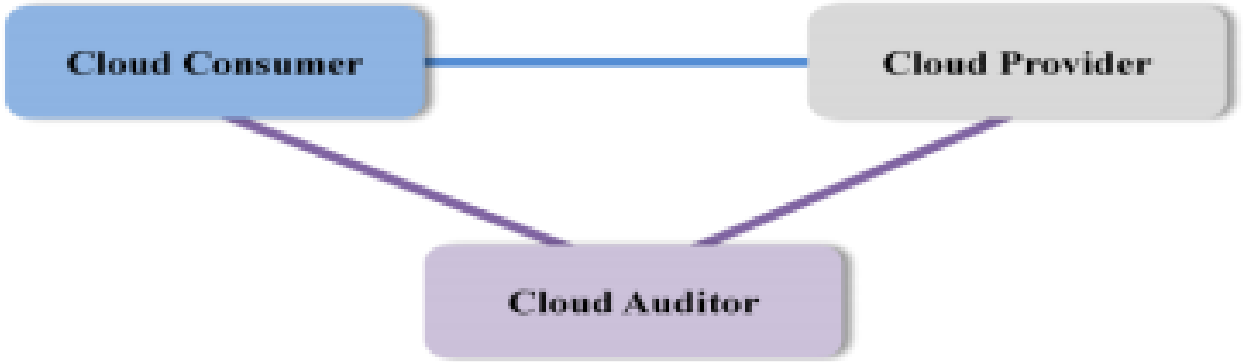
Scenarios in Cloud: 2

1. Cloud carriers provide the connectivity and transport of cloud services from cloud providers to cloud consumers.
2. Cloud provider participates in and arranges for two unique service level agreements (SLAs), one with a cloud carrier (e.g. SLA2) and one with a cloud consumer (e.g. SLA1).
3. A cloud provider may request cloud carrier to provide dedicated and encrypted connections to ensure the cloud services (SLA's).

NIST Cloud Computing Reference Architecture

Scenarios in Cloud: 3

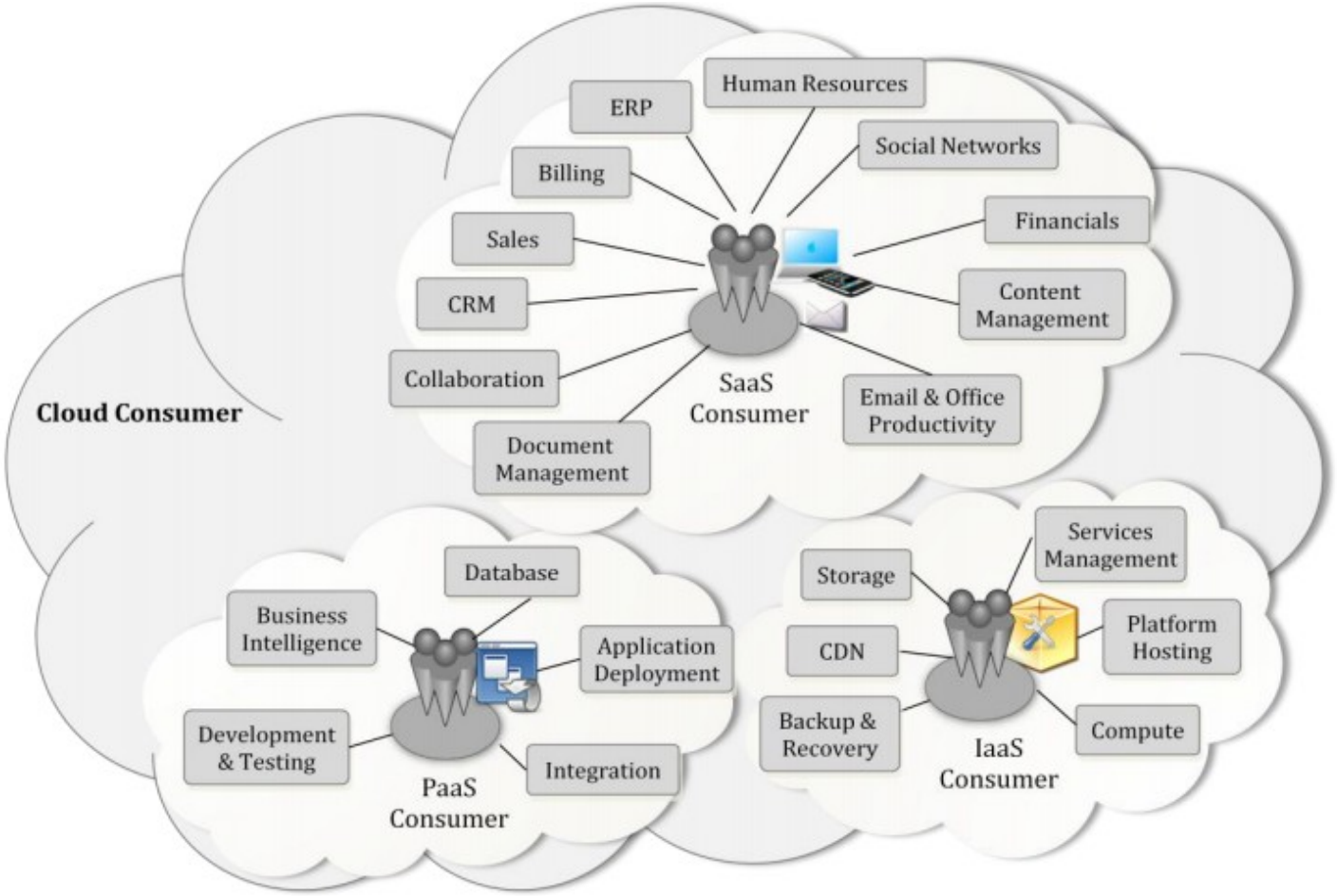
1. Cloud auditor conducts independent assessments for the operation and security of the cloud service.
2. The audit may involve interactions with both the Cloud Consumer and the Cloud Provider.



Cloud Consumer

- Cloud consumer browses & uses the service.
- Cloud consumer sets up contracts with the cloud provider.
- Cloud consumers need SLAs to specify the technical performance requirements fulfilled by a cloud provider.
- SLAs cover the quality of service, security, remedies for performance failures.
- A cloud provider list some SLAs that limit and obligate the cloud consumers by must acceptance.
- Cloud consumer can freely choose a cloud provider with better pricing with favorable conditions.
- Pricing policy and SLAs are non-negotiable.

Cloud Consumer



NIST Cloud Computing Reference Architecture

SaaS consumers

- SaaS consumers can be organizations that provide their members with access to software applications, end users who directly use software applications, or software application administrators who configure applications for end users.
- SaaS consumers can be **billed** based on the number of end users, the time of use, the network bandwidth consumed, the amount of data stored or duration of stored data.

PaaS consumers

- PaaS consumers can be application developers or administrators
  1. who design and implement application software
  2. application testers who run and test applications
  3. who publish applications into the cloud
  4. who configure and monitor application performance.
- PaaS consumers can be **billed** according to, processing, database storage and network resources consumed by the PaaS application, and the duration of the platform usage.

IaaS consumer

- IaaS consumer can be system developers, system administrators and IT managers who are interested in creating, installing, managing and monitoring services for IT infrastructure operations.
- IaaS consumer can be **billed** according to the amount or duration of the resources consumed, such as CPU hours used by virtual computers, volume and duration of data stored, network bandwidth consumed, number of IP addresses used for certain intervals.

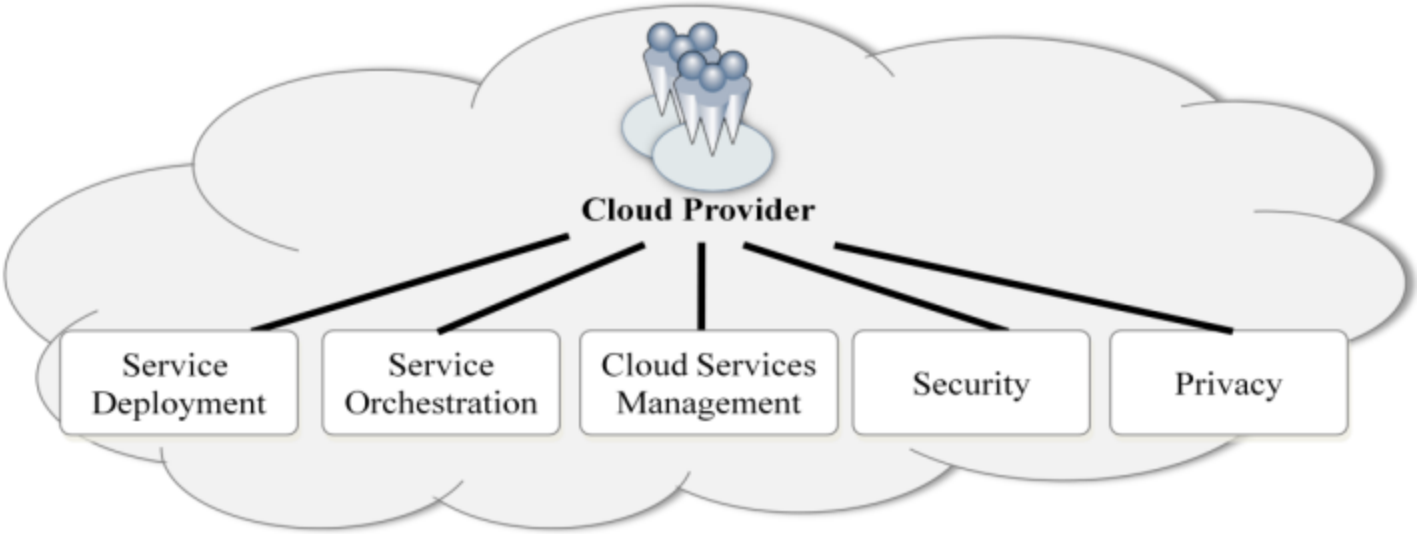
Cloud Provider

- Cloud Provider acquires and manages the computing infrastructure required for providing the services, runs the cloud software that provides the services, and makes arrangement to deliver the cloud services to the Cloud Consumers through network access.
- SaaS provider deploys, configures, maintains and updates the operation of the software applications on a cloud infrastructure. SaaS provider maintains the expected service levels to cloud consumers.
- PaaS Provider manages the computing infrastructure for the platform and components (runtime software execution stack, databases, and other middleware).
- IaaS Cloud Provider provides physical hardware and cloud software that makes the provisioning of these infrastructure services, for example, the physical servers, network equipments, storage devices, host OS and hypervisors for virtualization.

Cloud Provider

Five major activities of Cloud Provider's

- *Service deployment*
- *Service orchestration*
- *Cloud service management*
- *Security*
- *Privacy*



NIST Cloud Computing Reference Architecture

Cloud Auditor

- Audits are performed to verify conformance to standards.
- Auditor evaluates the security controls, privacy impact, performance, etc.
- Auditing is especially important for federal agencies.
- Security auditing, can make an assessment of the security controls to determine the extent to which the controls are implemented correctly, operating as intended, and producing the desired outcome. This is done by verification of the compliance with regulation and security policy.
- Privacy audit helps in Federal agencies comply with applicable privacy laws and regulations governing an individual's privacy, and to ensure confidentiality, integrity, and availability of an individual's personal information at every stage of development and operation.

Cloud Broker

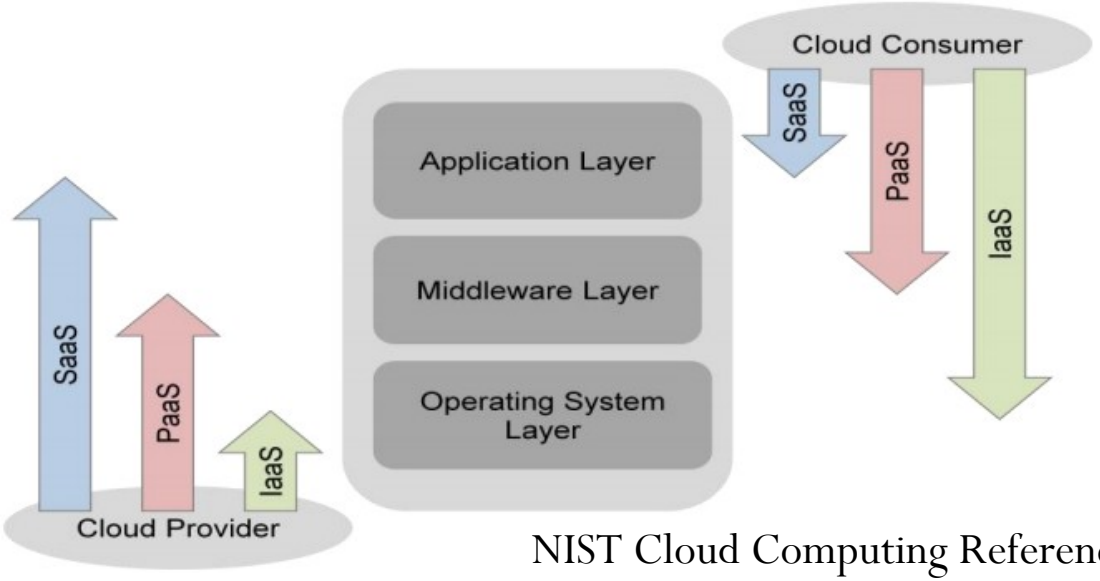
- Integration of cloud services can be complex for consumers. Hence cloud broker, is needed.
- Broker manages the use, performance and delivery of cloud services and negotiates relationships between cloud providers and cloud consumers.
- In general, a cloud broker can provide services in three categories:
  - *Service Intermediation*: Broker enhances a service by improving capability and providing value-added services to consumers. The improvement can be managing access to cloud services, identity management, performance reporting, enhanced security, etc.
  - *Service Aggregation*: Broker combines and integrates multiple services into one or more new services. The broker provides data integration and ensures the secure data movement.
  - *Service Arbitrage*: It is similar to service aggregation with the flexibility to choose services from multiple agencies. For example, broker can select service with the best response time.

Cloud Carrier

- Cloud carriers provide access to consumers through network, telecommunication and other access devices.
- For example, cloud consumers can obtain cloud services through network access devices, such as computers, laptops, mobile phones, mobile internet devices (MIDs), etc.
- The distribution of cloud services is normally provided by network and telecommunication carriers or a *transport agent*, where a transport agent refers to a business organization that provides physical transport of storage media such as high-capacity hard drives.
- Cloud provider can set up SLAs with a cloud carrier to provide services consistent with the level of SLAs offered to cloud consumers.

Scope of Control between Provider and Consumer

- Application layer are used by SaaS consumers, or installed/managed/ maintained by PaaS consumers, IaaS consumers, and SaaS providers.
- Middleware is used by PaaS consumers, installed/managed/maintained by IaaS consumers or PaaS providers. Middleware is hidden from SaaS consumers.
- IaaS layer is hidden from SaaS consumers and PaaS consumers. Consumers have freedom to choose OS to be hosted.



NIST Cloud Computing Reference Architecture

Various kind of Cloud computing architecture

Web 2.0, Mash-up, RSS Feed, Hadoop, HDFS, Storm, Various Google architectures, Eucalyptus & Ruby on Rails

https://sites.google.com/site/animeshchaturvedi07/academic-teaching/cloudcomputing

Web 2.0

Features

- Web 2 based Homepage
- Search
- RSS
- Chats, Sharing
- Personalized web sites,
- Mashups
- POD Casting, Video Casting, You Tube
- Focus on community, online collaboration, tagging

Mashup

- A mashup, is a phenomenon to create new services on web, with the combination of presentation, data, or functionality from two or more resources.
- What are the benefits?
  - Innovation, new business insights, increase agility, speed up the development, reduce development costs, easy & fast integration, widgets are readily available, reuse software, immediate benefit, less cost, rich ecosystem, may not be original raw source data, produce enriched results of data & services
- Mashup is divided into three layers with the following technologies.
- **Presentation:** HTML, XHTML, CSS, Javascript, Ajax, API.
- **Web Services:** are XMLHTTPRequest, XML-RPC, JSON-RPC, SOAP, REST.
- **Data:** Sending, storing and receiving. The technologies used are XML, JSON, KML.

RSS feed

- RSS is an XML language for syndicating subscribed contents on the web.
- Registers with an organization on web, who creates an RSS feed.
- User is automatically updated with the RSS feed (notification of new items)
- Information is received quickly at a location.

Hadoop

- "The Apache Hadoop project develops open source software for reliable, scalable, distributed computing"
- Created by Doug Cutting
- Named on his son's stuffed elephant

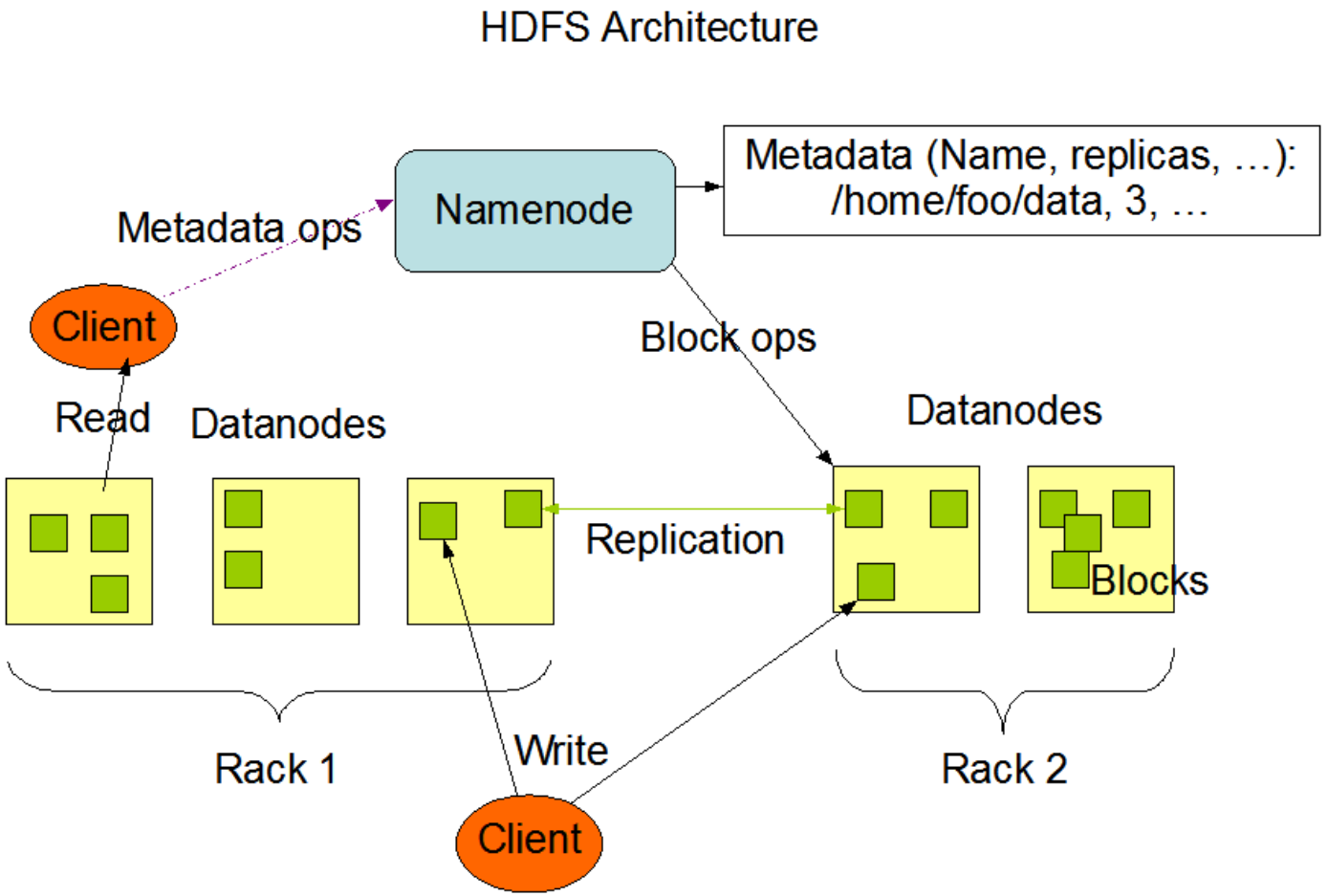
http://hadoop.apache.org/

Hadoop

- Software library and a framework.
- For distributed processing of large data sets across clusters of computers using simple programming models.
- Locality of reference
- Scalability: Scale up from single servers to thousands of machines, each offering local computation and storage.
  - Program remains same for 10, 100, 1000,… nodes
  - Corresponding performance improvement
- Fault-tolerant file system: Detect and handle failures, Delivering a highly-available.
  - **Hadoop Distributed File System (HDFS)**
  - Modeled on Google File system
- **MapReduce** for Parallel computation using
- Components – Pig, Hbase, HIVE, ZooKeeper

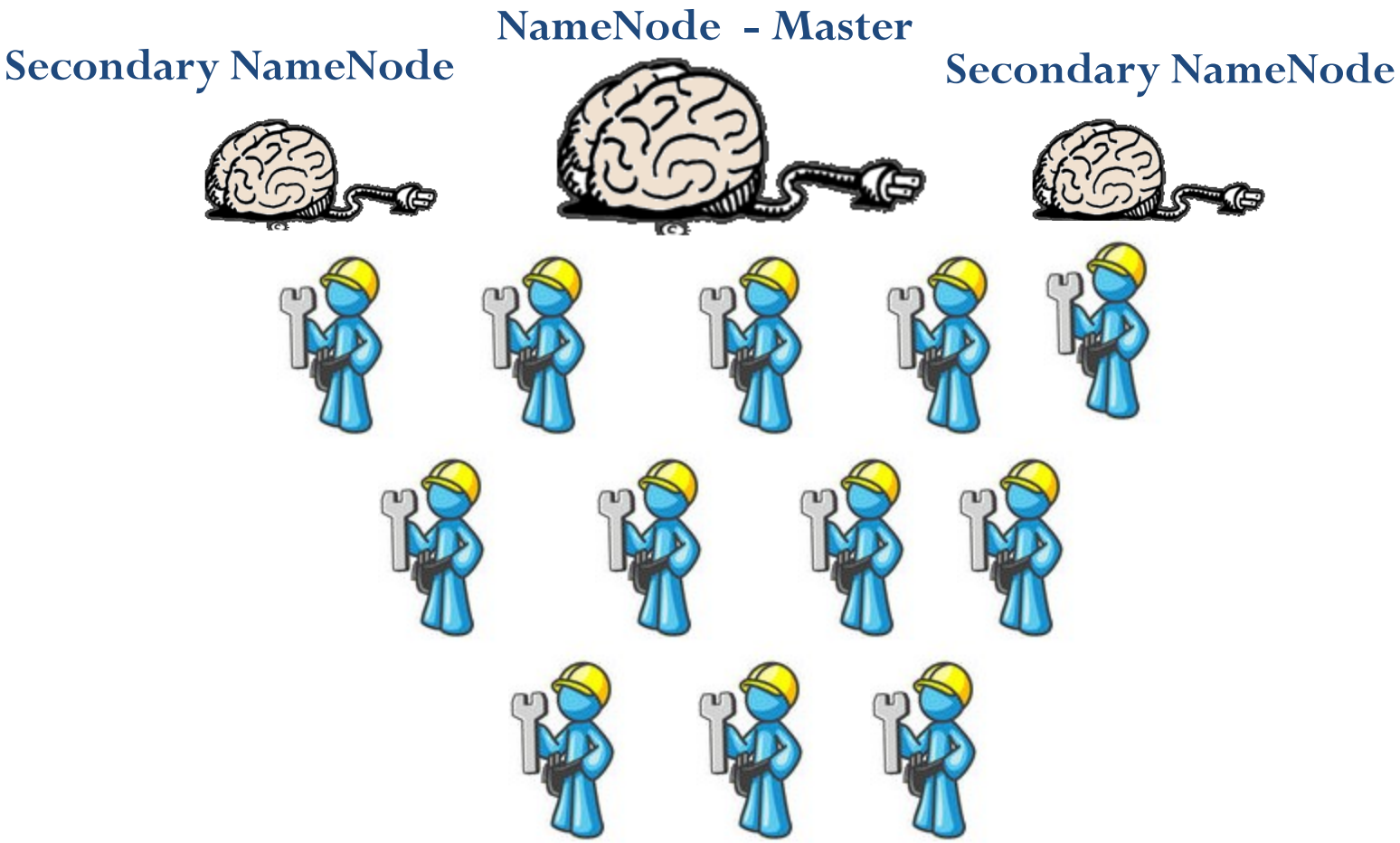
http://hadoop.apache.org/docs/r1.2.1/hdfs_design.html accessed on Jan 2015

Hadoop Distributed File System



http://hadoop.apache.org/docs/r1.2.1/hdfs_design.html accessed on Jan 2015

HDFS Force


NameNode - Master
Secondary NameNode
Secondary NameNode
DataNodes - Slave

Hadoop Distributed File System

Name Node: Manages File System - mapping files to blocks and blocks to data nodes. Maintains status of data nodes

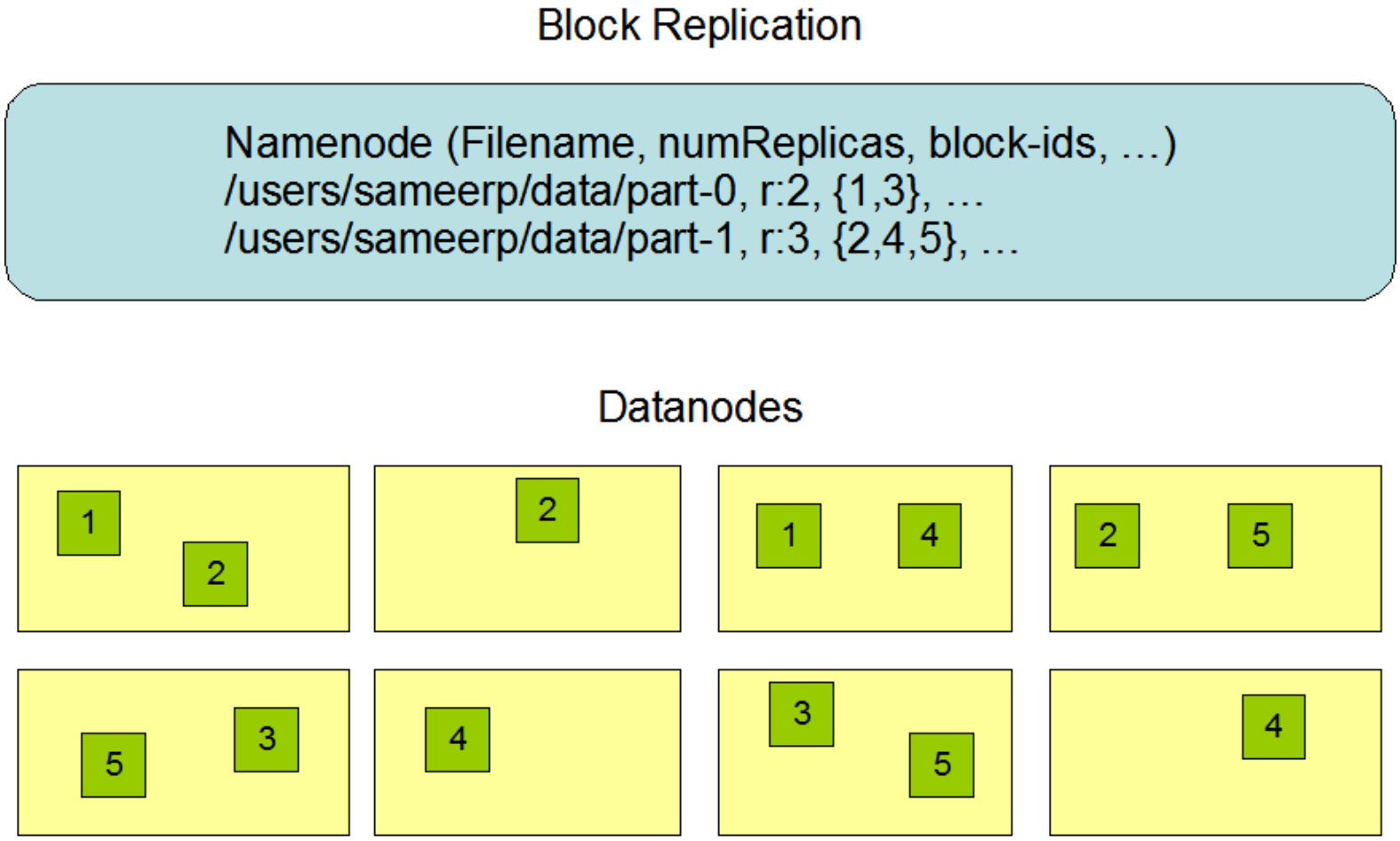
- Heartbeat: Datanode sends heartbeat at regular intervals, if heartbeat is not received, datanode is declared to be dead
- Blockreport: DataNode sends list of blocks on it. Used to check health of HDFS

Data Node:

- Replicates (On Datanode failure, On Disk failure and On Block corruption)
- Data integrity (Checksum for each block, Stored in hidden file)
- Rebalancing- balancer tool (addition of new nodes, decommissioning and deletion of some files)

http://hadoop.apache.org/docs/r1.2.1/hdfs_design.html accessed on Jan 2015

HDFS Data Replication

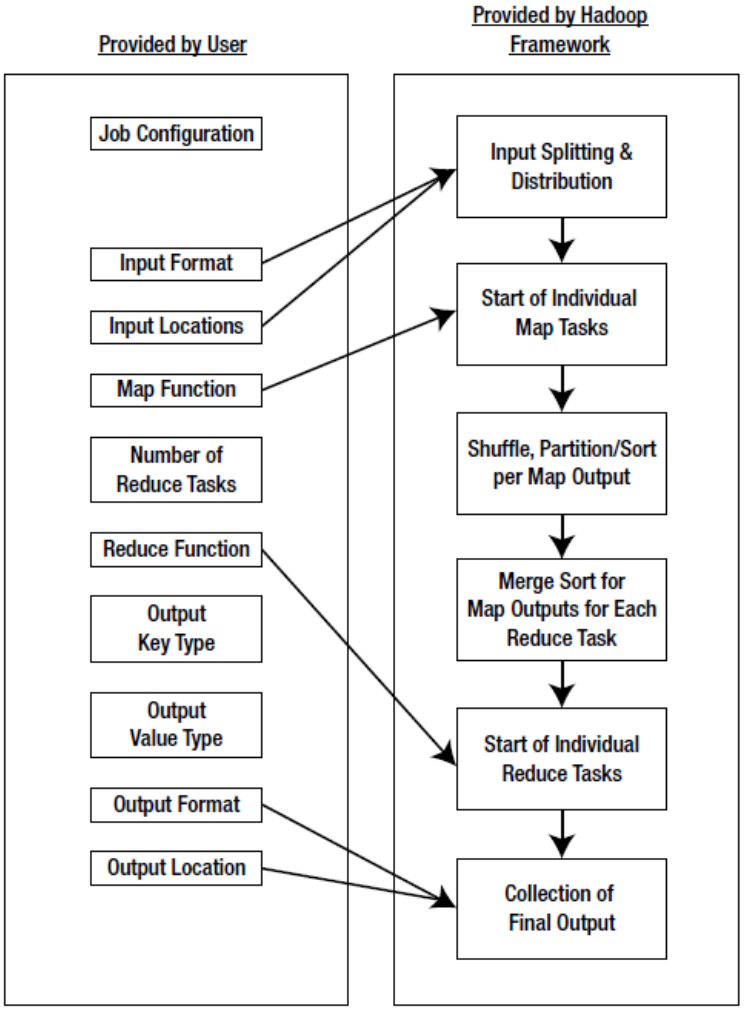


http://hadoop.apache.org/docs/r1.2.1/hdfs_design.html accessed on Jan 2015

Map Reduce

- Format of input- output (key, value)
  - Map:
    (k1, v1) → list (k2, v2)
  - Reduce:
    (k2, list v2) → list (k3, v3)

- MapReduce will not work for
  - Inter-process communication
  - Data sharing required
  - Example: Recursive functions

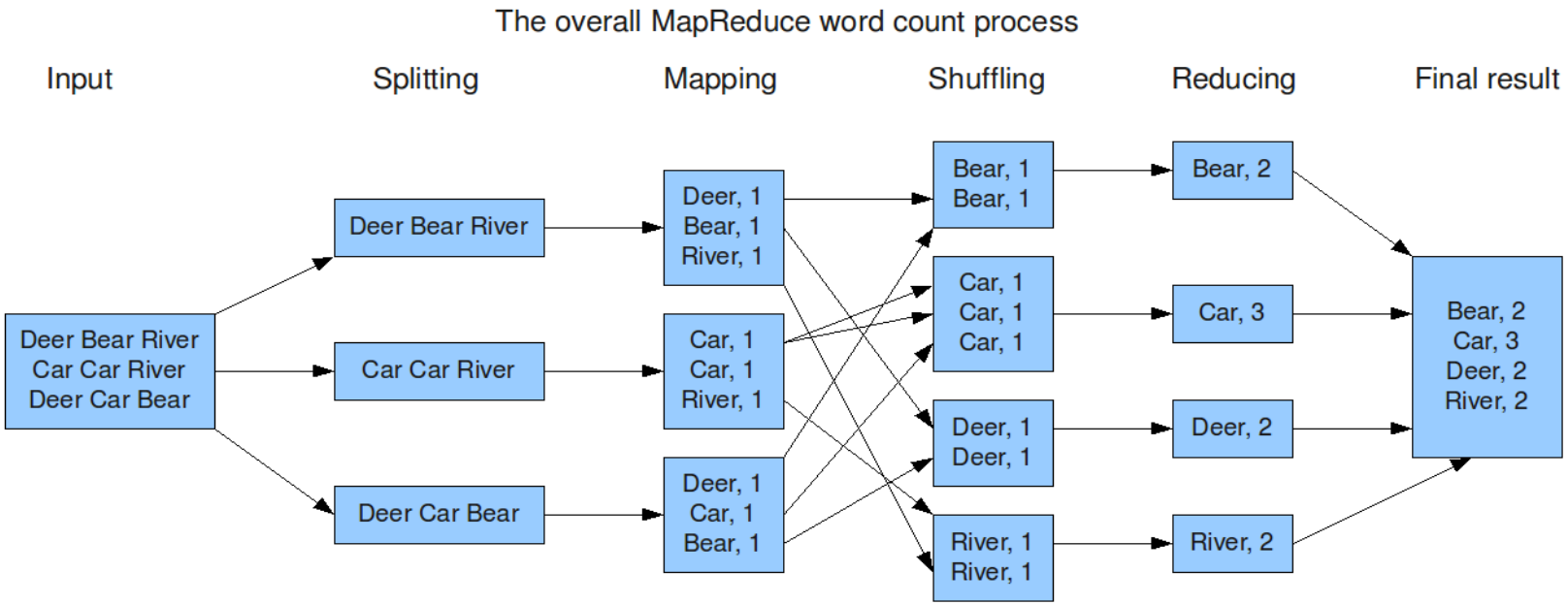


http://hadoop.apache.org/docs/r1.2.1/hdfs_design.html accessed on Jan 2015

Word Count

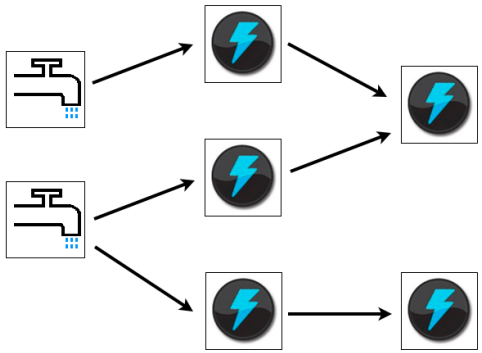
Map: Input lines of text to breaks them into words gives outputs for each word <key=word, value=1>

Reduce: Input <word, 1> output <word, + value>

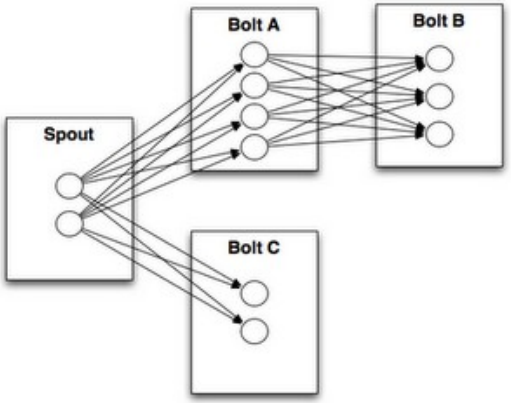


Source: Author: JocaPC
http://msacademic.rs/Blog.aspx?id=356

Strom



- Reliably for processing unbounded streams
- Hadoop for batch processing. Storm real-time processing
- Realtime analytics
- Online machine learning
- Continuous computation
- Distributed RPC.



- A million tupples can be processed per second per node. It is scalable, fault-tolerant, guarantees your data will be processed, and is easy to set up and operate.

https://storm.apache.org/

Storm

- **Topologies:** analogous to a MapReduce job. MapReduce job finishes, whereas a topology runs forever or until you kill it. A topology is a graph of spouts and bolts that are connected with stream groupings.
- **Streams:** is an unbounded sequence of tuples that is processed and created in parallel in a distributed fashion. Streams are defined with a schema that names the fields in the stream's tuples. Schema are integers, longs, shorts, bytes, strings, doubles, floats, booleans, and byte arrays. Every stream is given an id when declared.

https://storm.apache.org/documentation/Concepts.html

Storm

- **Spouts:** A spout is a source of streams in a topology. Generally spouts will read tuples from an external source and emit them into the topology.
  - Reliable Spouts: Replaying a tuple if it failed to be processed.
  - Unreliable Spouts. Forgets about the tuple as soon as it is emitted.
- **Bolts:** All processing in topologies is done in bolts. Bolts can do filtering, functions, aggregations, joins, talking to databases, and more. Bolts can do stream transformations into a new stream in a distributed and reliable way. Complex stream transformations often requires multiple steps and thus multiple bolts. For example, transform a stream of tweets into a stream of trending topics.

https://storm.apache.org/documentation/Concepts.html

Google File System

- A scalable distributed file system for large distributed data-intensive applications.
- It provides fault tolerance. High aggregate performance to a large number of clients.
- It is widely deployed within Google as the storage platform for the generation and processing of data used by Google service as well as research and development efforts that require large data sets.
- The largest cluster to date provides hundreds of terabytes of storage across thousands of disks on over a thousand machines, and it is concurrently accessed by hundreds of clients. Earlier Google effort, "BigFiles", developed by Larry Page and Sergey Brin. Files are divided into fixed-size chunks of 64 megabytes.
- GFS is not implemented in the kernel of an operating system, but is instead provided as a userspace library.

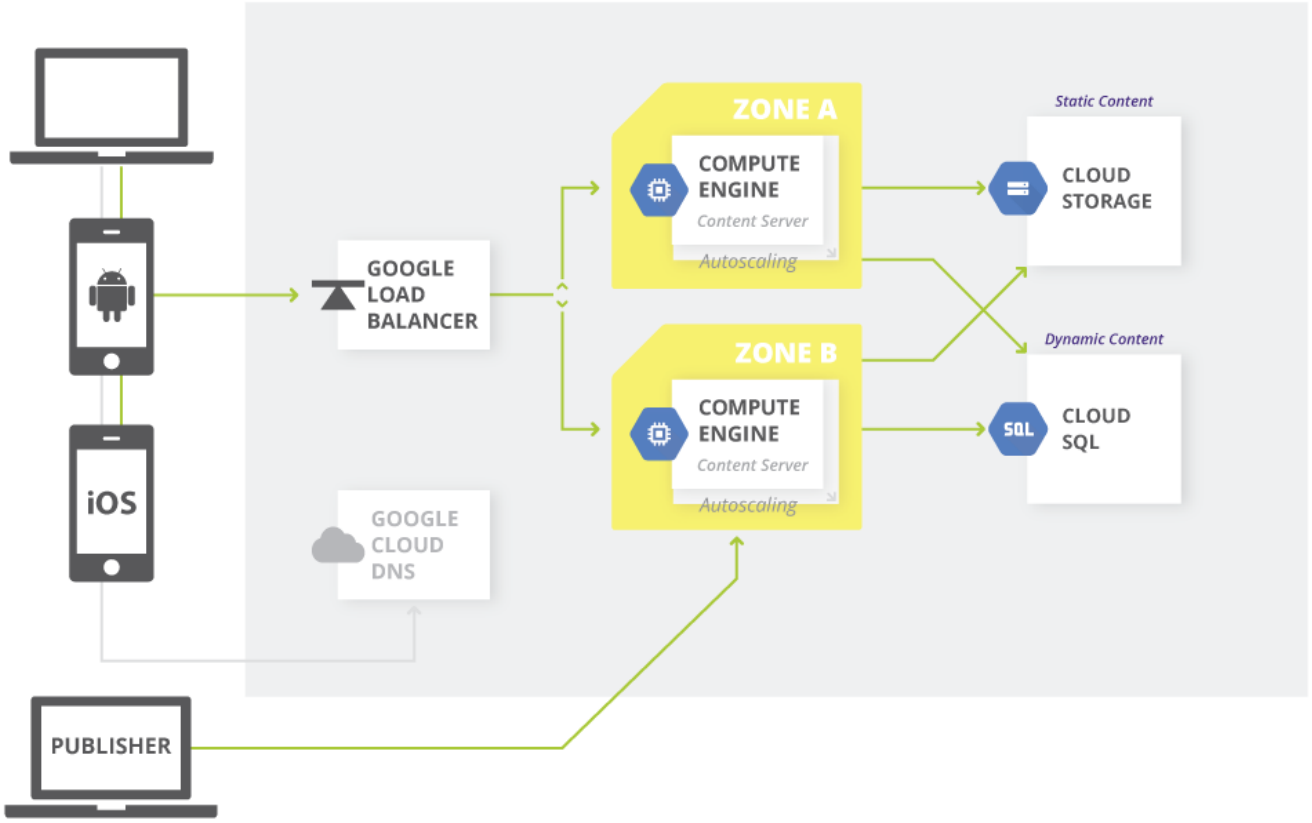
http://research.google.com/archive/gfs.html
http://en.wikipedia.org/wiki/Google_File_System

Google Content Management

- Web information, marketing campaigns or social media.
- Personalized for individual users or groups.
- Google Cloud Platform (GCP) components and services to create a Content Management system.
- Google Load Balancer to support traffic routed to multiple zones for high availability.
- Google's Cloud DNS provides a robust DNS manages the domain.
- Static content → Cloud Storage
- Dynamic content→ Cloud SQL implementation.

https://cloud.google.com/solutions/architecture/contentmanagement accessed 15 Jan 2015

Google Content Management



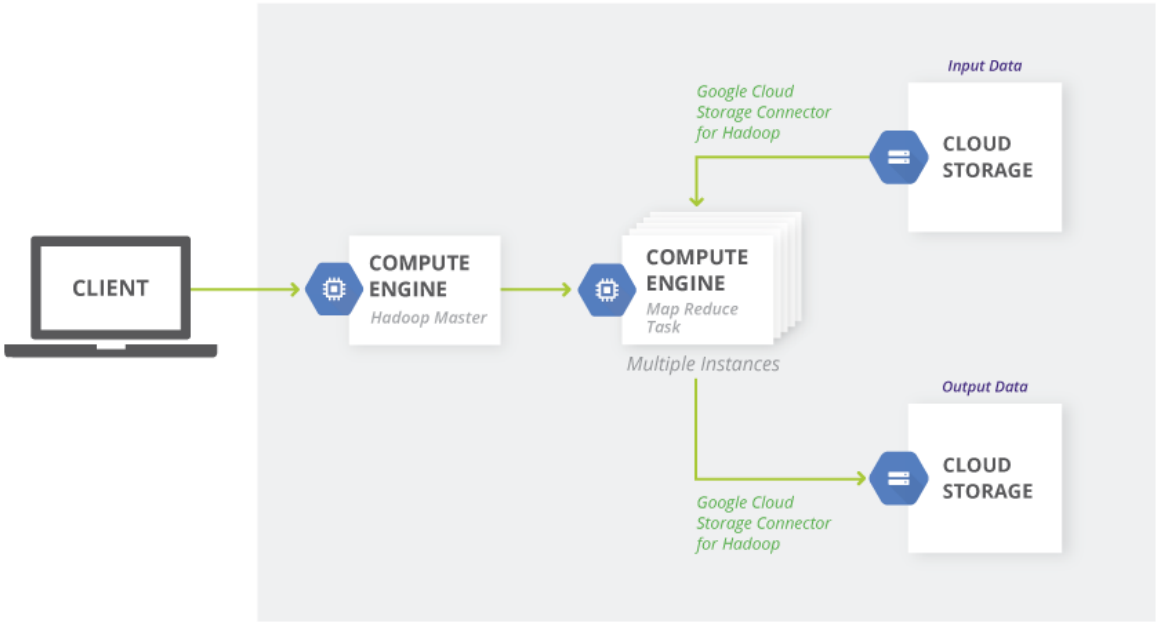
https://cloud.google.com/solutions/architecture/contentmanagement accessed 15 Jan 2015

Architecture: Hadoop on Google Cloud Platform

- Infrastructure for MapReduce using Hadoop.
- Compute power and Cloud Storage to store the input and output of the MapReduce jobs.
- Hadoop Master: includes the HDFS NameNode and the MapReduce JobTracker.
- Nodes in the cluster will run MapReduce tasks with DataNode and MapReduce TaskTracker.
- Backing-up the storage through Google Cloud Storage Connector for Hadoop. HDFS, can be used, Google's Cloud Storage.

https://cloud.google.com/solutions/architecture/hadoop accessed 15 Jan 2015

Architecture: Hadoop on Google Cloud Platform



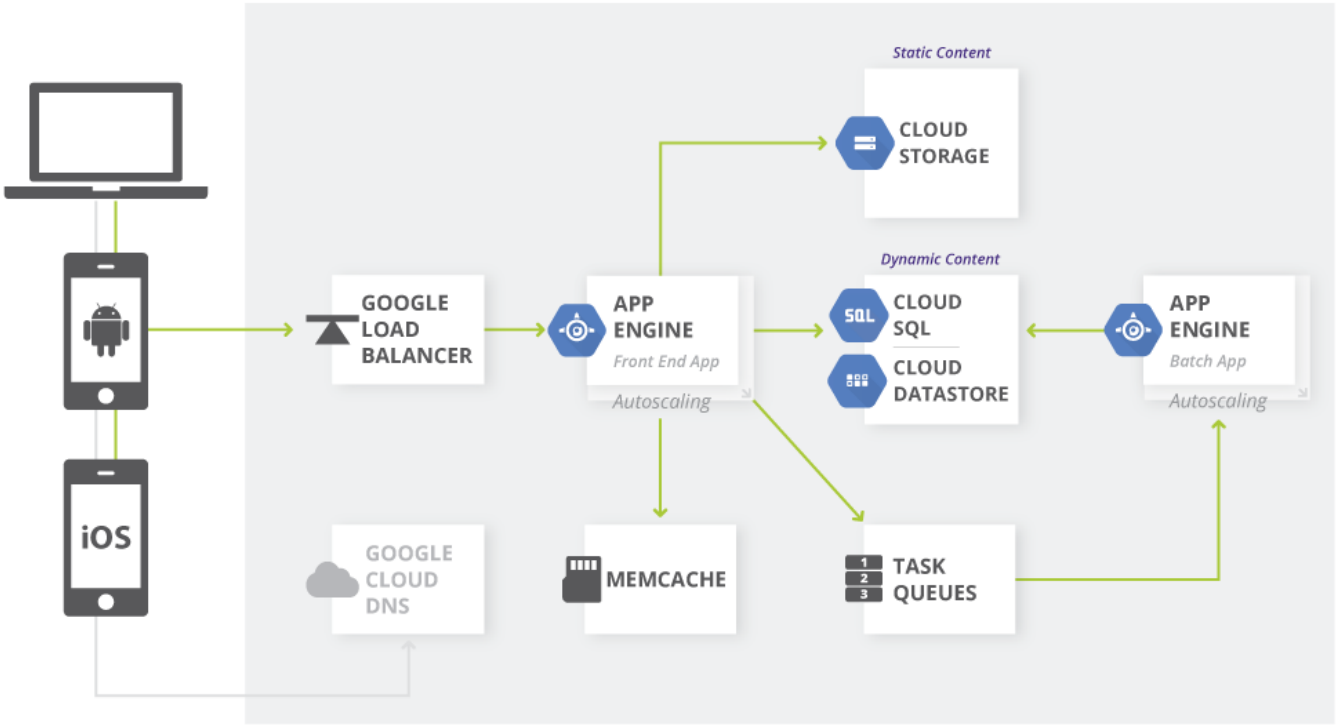
https://cloud.google.com/solutions/architecture/hadoop accessed 15 Jan 2015

Architecture: Web Application on Google App Engine

- Simple development and deployment of Web Applications with autoscaling compute power as well as the integrated features like distributed in-memory cache, task queues and datastore, to create robust applications quickly and easily.
- For applications written in Java, Python, PHP and Go.
- Supports multiple application versions which support A/B testing.
- Memcache is an in-memory cache to provide extremely high speed access to information cached by the web server (e.g. authentication or account information).
- Task Queues provide a mechanism to offload longer running tasks to backend servers, freeing the front end servers to service new user requests.
- Google Load Balancer which provides transparent load balancing to applications.
- Google's Cloud DNS is used to manage DNS domain of user.

https://cloud.google.com/solutions/architecture/webapp accessed 15 Jan 2015

Architecture: Web Application on Google App Engine



https://cloud.google.com/solutions/architecture/webapp accessed 15 Jan 2015

Google Web Tool - Kit

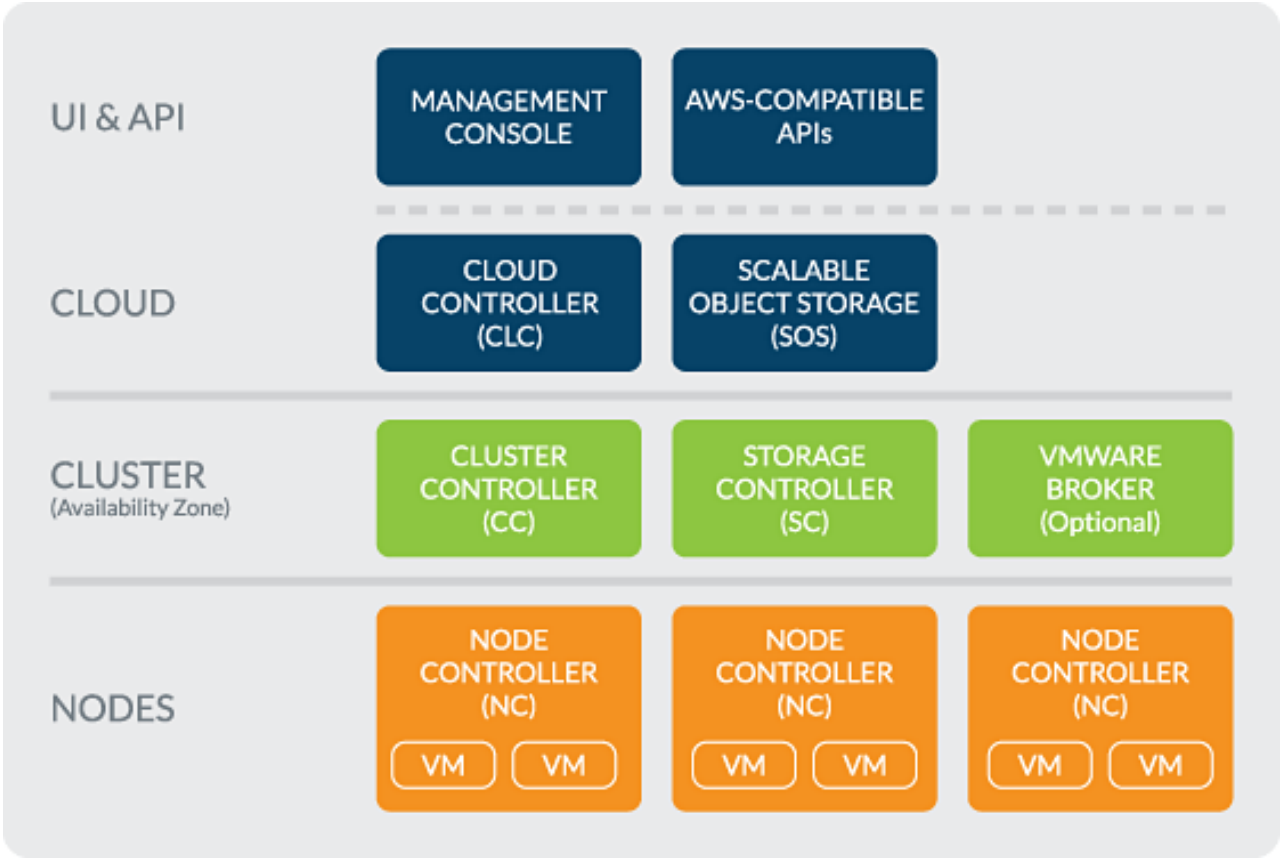
- GWT gives us API to design rich web applications.
- GWT is a Swing-like Java framework. Developer can write web application without writing HTML or JavaScript code.
- GWT is a development environment similar to any Web-Server-Code or Desktop-app development environment.
- GWT helps to debug, re-factor and unit test a Web-Client.
- GWT provides a so-called hosted mode, that allows developers to debug Java code, as well as a web mode which executes the GWT-generated JavaScript code.
- **Google uses GWT for its Sites:** Google Docs, Google AdSense, Google Wallet
- **Other Sites:** gogrid.com, Scenechronize, Google Moderator, Whirled. See more at http://gwtgallery.appspot.com/

Eucalyptus Cloud

- Eucalyptus **cloud** is highly scalable.
- **Eucalyptus Cloud Components:** The six components are grouped into three separate levels.
- Cloud Level: The Cloud level of the computing architecture is comprised of only two components and while used by many users, the transactions at each component are typically small.
  - Cloud Controller (CLC)
  - Scalable Object Storage (SOS)
- Cluster Level (i.e., Availability Zone)
  - Cluster Controller (CC)
  - Storage Controller (SC)
  - VMware Broker (Optional)
- Node Level: The Node level may have many components, but each component only supports a few users, even though the transactions are larger. This **distributed cloud architecture** is flexible enough to support businesses of any size.
  - Node Controller (NC)

https://www.eucalyptus.com/eucalyptus-cloud/iaas/architecture

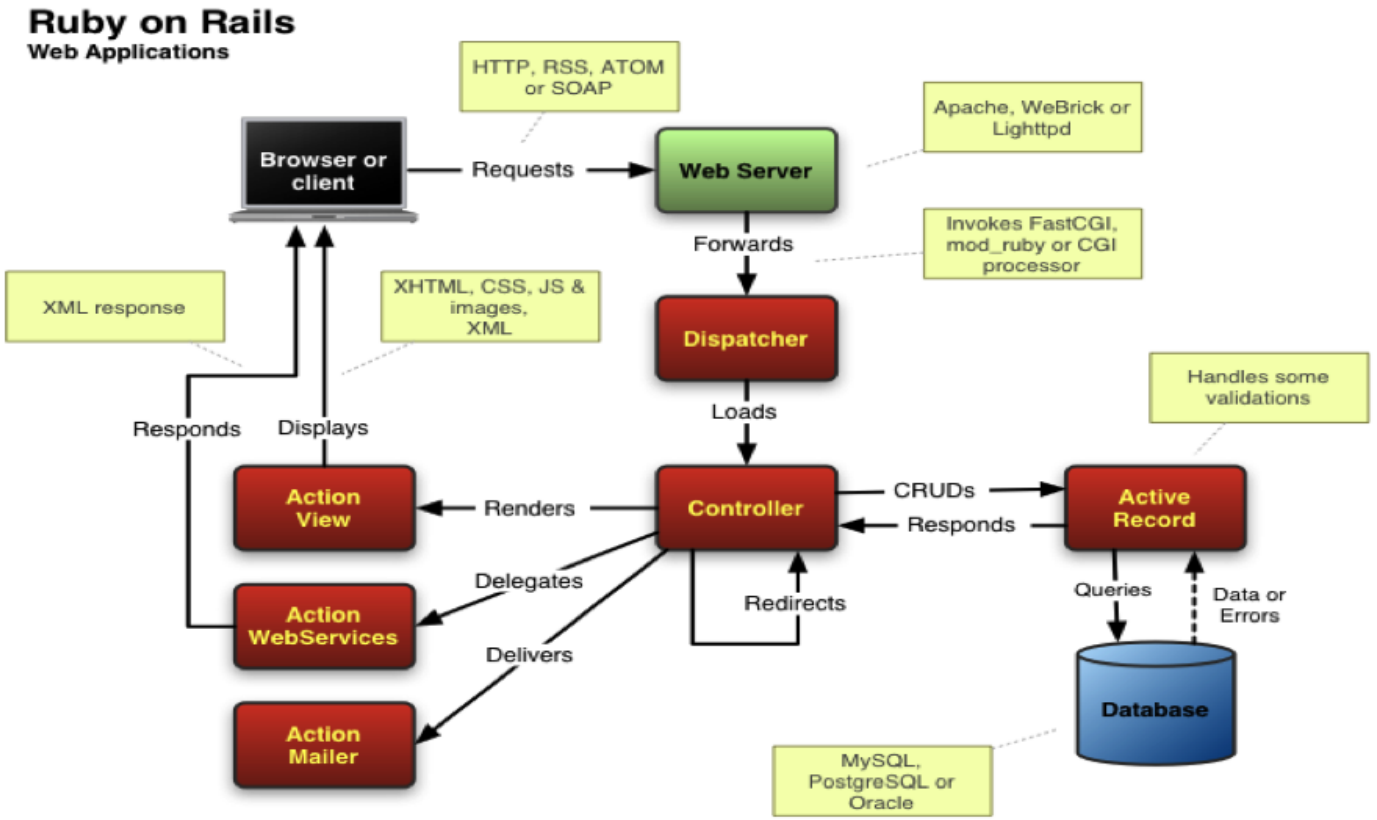
Eucalyptus Cloud Architecture



https://www.eucalyptus.com/eucalyptus-cloud/iaas/architecture

Ruby on Rails

- A web applications development framework. Very popular in the agile development with the help of convention over configuration management. Ruby on Rails uses the Model-View-Controller (MVC)



http://adrianmejia.com/blog/2011/08/11/ruby-on-rails-architectural-design/

Ruby on Rails

RESTful Architecture

- Representational State Transfer (REST) is an alternative to web services, such as SOAP and WSDL.
- Works on HTTP protocol for operations: Create, Read, Update and Delete (CRUD).
- RESTful is useful when it is required stateless, limited bandwidth (specially for mobile devices no overhead of SOAP), and when service provider and user have complete information of operations.

Ruby on Rails

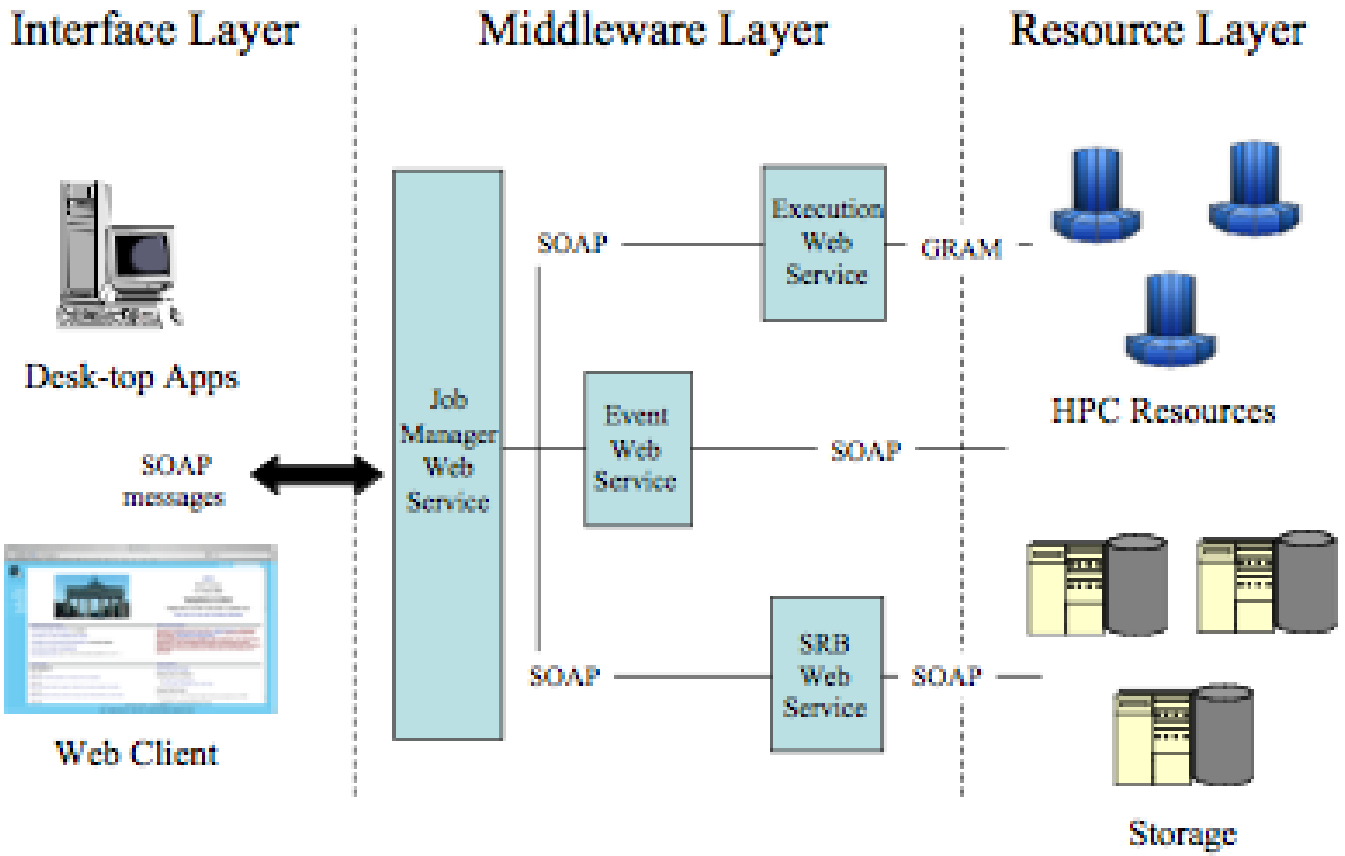
- MVC separates business logic from HTML views. Architectural pattern in order to improve the maintainability of the application.
- **Model:** Carries the business logic and rules to manipulate the data.
  - Models represent the information in the database.
  - Manages interaction with database.
- **View:** Front-end of the application, representing the user interface.
  - HTML files with embedded Ruby code. Used to display data in the form of views
  - Formats, such as HTML, PDF, XML, RSS and more.
- **Controller:** it interact with models and views.
  - Incoming requests are processed by the controllers.
  - Controller process the data from the models and pass it to the views for presentation.

SOA, Grid and Cloud Computing

How Services are being involved in Grid and Cloud computing architecture.

https://sites.google.com/site/animeshchaturvedi07/academic-teaching/cloudcomputing

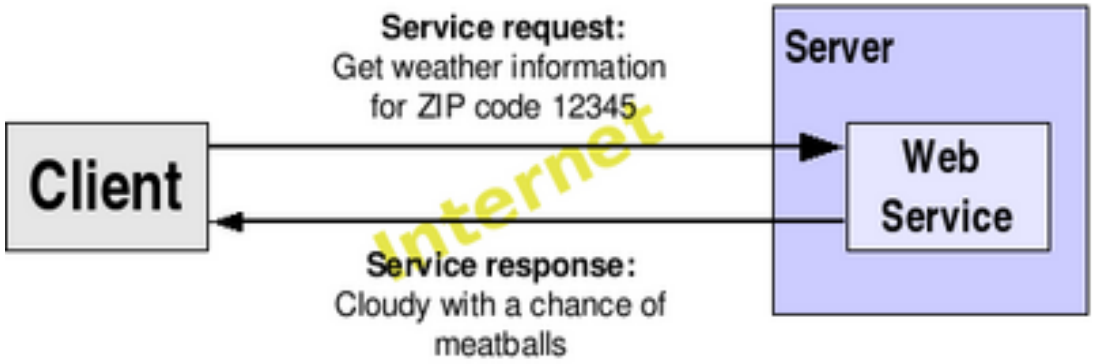
Common Architecture



Borja Sotomayor, "The Globus Toolkit 3 Programmer's Tutorial,"

URL V/S URI

- Websites for humans, Web Services for software
- http://webservices.mysite.com/weather/us/WeatherService
- Web Services use HTTP as the application layer protocol to exchange SOAP messages over the network.

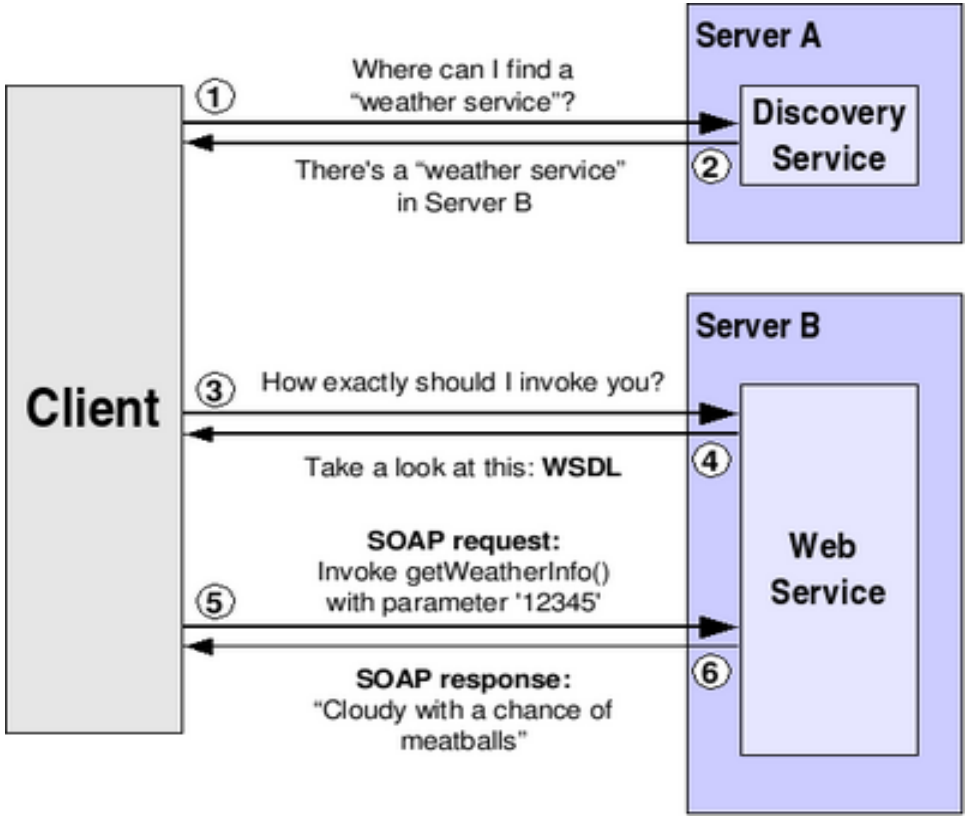


Borja Sotomayor, "The Globus Toolkit 3 Programmer's Tutorial,"

Web Service Invocation

Sequence of Web service invocation process

- Discovery service
- Discovery service will reply
- Web Service Descriptive Language (WSDL)
- SOAP request
- Method invocation
- SOAP response

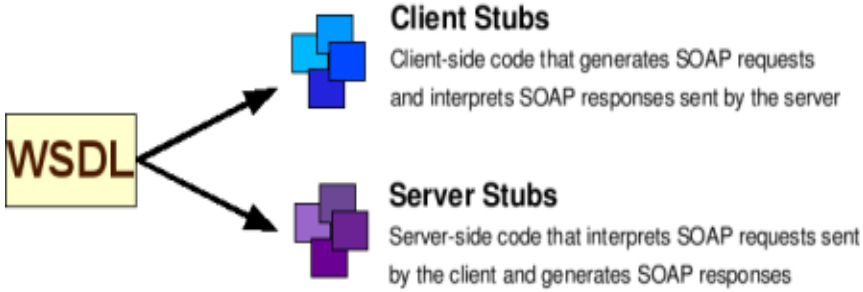


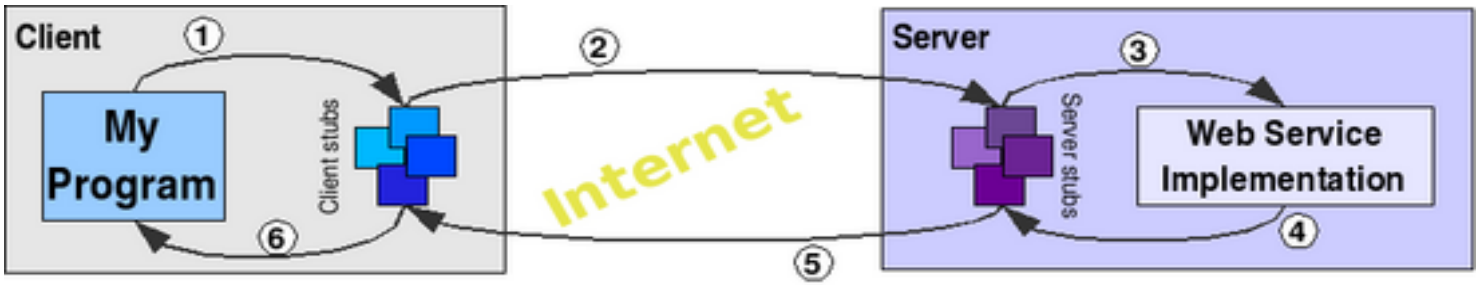
Borja Sotomayor, "The Globus Toolkit 3 Programmer's Tutorial,"

Web Service Invocation

A stub is an API that handles Marshalling/Unmarshalling. Marshalling also refers to serializing and Unmarshalling also refers to deserializing.

1. SOAP request then Marshalling or serializing
2. HTTP protocol
3. Unmarshaling or deserializing then SOAP request
4. SOAP response then again Marshalling
5. HTTP protocol
6. Unmarshaling then SOAP response

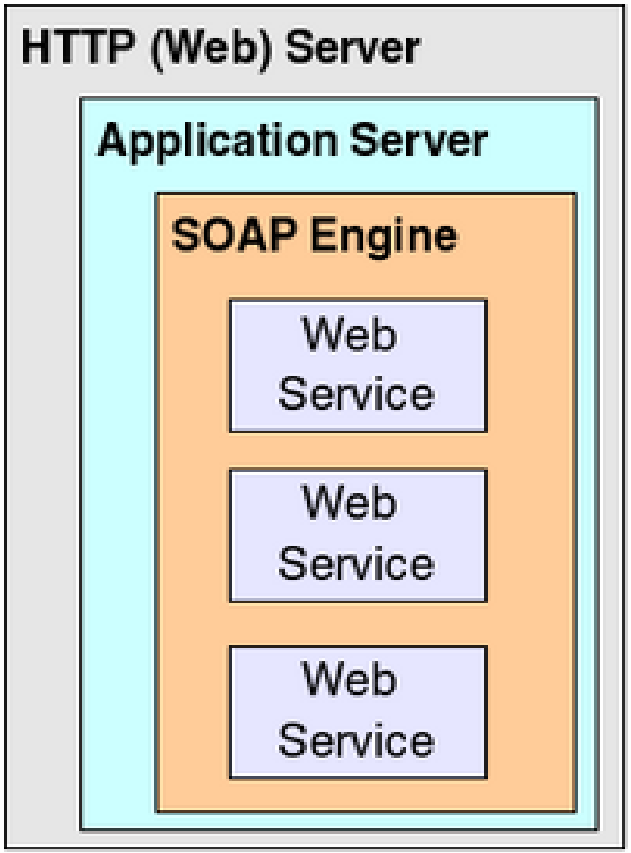




Borja Sotomayor, "The Globus Toolkit 3 Programmer's Tutorial,"

Web Service Invocation

- Web service
- SOAP engine
- Application server
- HTTP Server



Borja Sotomayor, "The Globus Toolkit 3 Programmer's Tutorial,"

Grid Architecture

- *Fabric layer* grids provide access to different resource types such as compute, storage and network resource, code repository, etc.
- *Connectivity layer* defines core communication and authentication protocols for easy and secure network transactions.
- *Resource layer* defines protocols for the publication, discovery, negotiation, monitoring, accounting and payment of sharing operations on individual resources.
- *Collective layer* captures interactions across collections of resources, directory services such as MDS (Monitoring and Discovery Service)
- *Application layer* comprises whatever user applications built on top of the above protocols and APIs and operate in VO environments.

Foster, Ian, et al. "Cloud computing and grid computing 360-degree compared." *Grid Computing Environments Workshop, 2008. GCE'08*. IEEE, 2008.

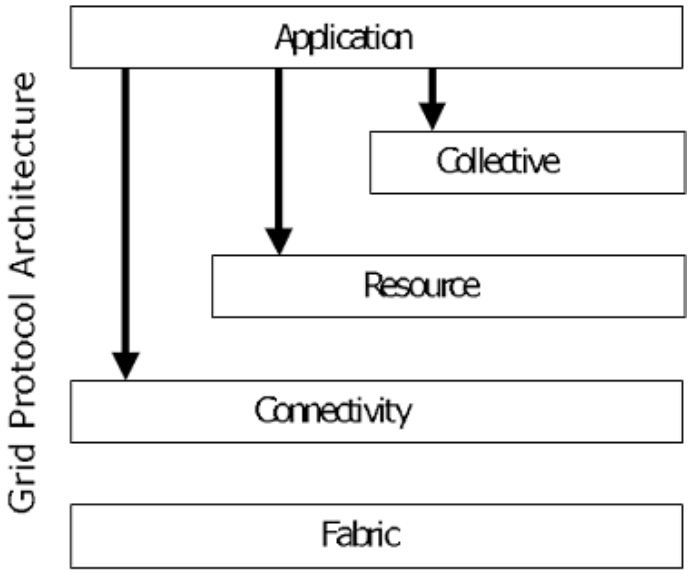
Cloud Architecture

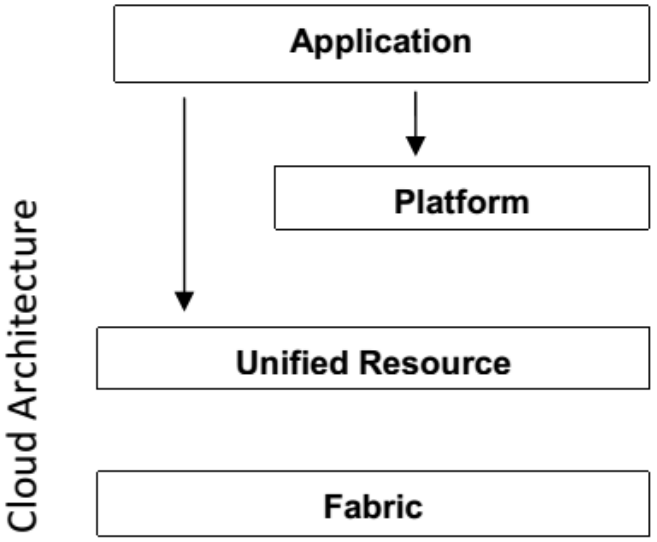
- *Fabric layer* contains the raw hardware level resources, such as compute resources, storage resources, and network resources.
- *Unified resource layer* contains resources that have been abstracted/encapsulated (usually by virtualization) so that they can be exposed to upper layer and end users as integrated resources, for instance, a virtual computer/cluster, a logical file system, a database system, etc.
- *Platform layer* adds on a collection of specialized tools, middleware and services on top of the unified resources to provide a development and/or deployment platform. For instance, a Web hosting environment, a scheduling service, etc.
- *Application layer* contains the applications that would run in the Cloud

Foster, Ian, et al. "Cloud computing and grid computing 360-degree compared." *Grid Computing Environments Workshop, 2008. GCE'08*. IEEE, 2008.

Grid and Cloud Architecture

- All the layers are being influenced with the fundamental concepts of *Services* technology.





Foster, Ian, et al. "Cloud computing and grid computing 360-degree compared."*Grid Computing Environments Workshop, 2008. GCE'08*. IEEE, 2008.

Transactional, On Demand & Distributed Computing

https://sites.google.com/site/animeshchaturvedi07/academic-teaching/cloudcomputing

CAPS Theorem

- Also known as Brewer's theorem

"It is impossible for a distributed computer system to simultaneously provide following three together with guarantees in single instance"

- **Consistency:** all nodes can view the same data at the same time
- **Availability:** a guarantee that every request will receives response for success or failure
- **Partition:** the system continues to operate irrespective of the loss or failure of a node.

No SQL

- "Not Only SQL" i.e. database systems that works on other than the tabular relations used in relational databases.
- No SQL systems may also support SQL-like query languages.
- No SQL database systems has data structures different from relational databases (key-value, graph, document).
- High usage in big data and real-time web applications.
- Mostly NoSQL databases systems follows the CAP theorem.
- Huddle in adoption of NoSQL stores are low-level query languages, lack of standardized interfaces, and huge investments in existing SQL.
- Some NoSQL systems do not provides all four ACID properties together (atomicity, consistency, isolation and durability).

Categories of NoSQL databases

- **Column Oriented:** Accumulo, Cassandra, Druid, HBase, Vertica
- **Document-Oriented:** Clusterpoint, Apache, CouchDB, Couchbase, MarkLogic, MongoDB, OrientDB
- **Key-value:** Dynamo, FoundationDB, MemcacheDB, Redis, Riak, FairCom c-treeACE, Aerospike, OrientDB
- **Graph:** Allegro, Neo4J, InfiniteGraph, OrientDB, Virtuoso, Stardog
- **Multi-model:** OrientDB, FoundationDB, ArangoDB, Alchemy Database, CortexDB.

http://en.wikipedia.org/wiki/NoSQL

On Demand Computing

- It is one of required feature of Cloud Computing.
- Enterprise-level computing model
- Technology and computing resources are allocated to the organization or its individual users on the basis of demand.
- Example of Computing resources are
  CPU cycles, bandwidth availability, storage and applications c.
- ODC is needed for: Bandwidth-heavy applications, big data computing, to collect the big data.
- On-demand computing also referred to as utility computing. Thus most of the concept of Cloud Computing follows here such as Pay-as-you-go, HPC, infinite illusion resources.

Distributed Computing

- Field that studies distributed systems.
- Software system in which components located on networked computers communicate and coordinate by using and passing messages.
- Grid, Utility and Cloud computing are the branch of Distributed Computing.
- Fundamentally based on the Remote Procedure Call (RPC), message queues and location transparency.

References

- By Foster, Y. Zhau, R. Ioan, and S. Lu. "Cloud Computing and Grid Computing: 360-Degree Compared." Grid Computing Environments Workshop, 2008.
- http://en.wikipedia.org/wiki/CAP_theorem
- Liu, F., Tong, J., Mao, J., Bohn, R., Messina, J., Badger, L., & Leaf, D. (2011). NIST Cloud Computing Reference Architecture. *NIST Special Publication*, *500*, 292.
- http://hadoop.apache.org/
- http://hadoop.apache.org/docs/r1.2.1/hdfs_design.html
- http:// research.google.com/archive/gfs.html
- https://cloud.google.com/solutions/architecture/
- https://www.eucalyptus.com/eucalyptus-cloud/iaas/architecture
- https://storm.apache.org/documentation/Concepts.html
- http://adrianmejia.com/blog/2011/08/11/ruby-on-rails-architectural-design/
- http://en.wikipedia.org/wiki/CAP_theorem
- http://en.wikipedia.org/wiki/NoSQL
- http://en.wikipedia.org/wiki/Distributed_computing
- http://www.webopedia.com/TERM/O/ODC.html

https://sites.google.com/site/animeshchaturvedi07/academic-teaching/cloudcomputing

Note: All links were accessed in January 2015

ขอบคุณ

Thai

Grazie

Italian

תודה רבה

Hebrew

Gracias

Spanish

Спасибо

Russian

Thank You

Obrigado

Portuguese

شكراً

Arabic

English

Merci

French

多謝

Traditional Chinese

धन्यवाद

Hindi

Danke

German

https://sites.google.com/site/animeshchaturvedi07/

多谢

Simplified Chinese

நன்றி

Tamil

Tamil

ありがとうございました

Japanese

감사합니다

Korean

ขอบคุณ

Thai

Grazie

Italian

תודה רבה

Hebrew

Gracias

Spanish

Спасибо

Russian

Thank You

Obrigado

Portuguese

شكراً

Arabic

English

Merci

French

多謝

Traditional Chinese

धन्यवाद

Hindi

Danke

German

https://sites.google.com/site/animeshchaturvedi07/

多谢

Simplified Chinese

நன்றி

Tamil

Tamil

ありがとうございました

Japanese

감사합니다

Korean