



ISSN 2047-3338

X-as-a-Service: Cloud Computing with Google App Engine, Amazon Web Services, Microsoft Azure and Force.com

Rabi Prasad Padhy¹, Manas Ranjan Patra² and Suresh Chandra Satapathy³

¹Oracle Corporation, Bangalore-29, India

²Dept. of Computer Science, Berhampur University, Berhampur, India

³Dept. of Computer Science & Engineering, ANITS, Sanivasala, India

¹rabi.padhy@gmail.com, ²mrpatra12@gmail.com, ³sureshsatapathy@ieee.org

Abstract—Internet has become pervasive in our daily life and Cloud computing is an emerging paradigm where computing resources are offered over the Internet as scalable, on-demand (Web) services. A company deploying an Internet service needs to invest huge amounts of money on infrastructure needs to serve possible users which is not a problem for large enterprises but when it comes to Small and Medium Enterprises (SME) affordability becomes a huge factor with the huge infrastructure come problems like machines failure, hard drive crashes, software bugs, etc. This might be a big problem for such a community. Cloud Computing is the ultimate solution to this problem. Rather than buying, installing and operating its own systems, an organization can rely on a cloud provider to do this for them. Cloud Computing key market leaders like Google, Amazon, Microsoft and Salesforce.com introduces new operating and business models that allow customers to pay for the resources they effectively use, instead of making heavy upfront investments. The aim of this paper is to provide the concepts as well as the technology behind Cloud Computing in general and analyze most popular platforms, The Google App Engine, Amazon Web Services, Windows Azure Platform and Force.com. This paper also discussed to identify companies for their favorability in the migration to the Cloud Architecture.

Index Terms—Cloud Computing, Cloud Architecture, Cloud Platforms, Google App Engine, SaaS, PaaS, IaaS and Cloud Database

I. INTRODUCTION

THE future of computing lies in cloud computing, whose major goal is reducing the IT Infrastructure costs while increasing processing throughput and decreasing CPU time, increasing reliability, availability and flexibility. Cloud computing is Internet based system development in which large scalable computing resources are provided “as a service” over the Internet to users. Cloud computing is built on top of diverse technologies like virtualization, distributed computing, utility computing, and more recently networking, web and software services. The concept of cloud computing incorporates web infrastructure, assorted services (IaaS, PaaS, SaaS), models and many technologies attracted more and more attention from industry and research community [1].

Fig. 1 shows a high level view of cloud computing Infrastructure. In cloud computing, the resources hosted within clouds can be anything: they could be database services, virtual servers (virtual machines), complete service workflows or complex configurations of distributed computing systems such as clusters [2]. Regardless of their nature, all resources are provided via services to clients (users or software processes) by computers rented from the cloud (such as those offered by e.g., Amazon, Google, Microsoft), rather than by private systems. The services are provided on demand and clients only pay for the quantity of resources (data storage, computation, etc.) they use [3]. In addition to services and resources, cloud computing has providers of two forms: service providers and cloud providers. A cloud provider is the entity that offers and maintains a cloud and may offer internally developed services on the cloud. A service provider is an entity that creates and maintains services that are published in and ran on clouds. For example, a service provider may not have the capita to host its own services hence rents space on a cloud to reduce costs. The best known cloud service providers like Google, Amazon, Microsoft and Salesforce.com.

II. CLOUD COMPUTING: SERVICE DELIVERY MODELS

Clouds computing services fall into three categories: applications (Software as a Service i.e. SaaS), platform (Platform as a Service i.e. PaaS) and hardware (Infrastructure as a Service i.e. IaaS). These offered services can be accessed anytime, anywhere in the world over the internet [4].

Software as a Service: In the SaaS category, there is delivery of use-specific services over the Internet (such as CRM software and email). The benefit of SaaS clouds is that clients only focus on the use of the software and do not have to worry about the cost and effort to keep software licenses current nor the handling of software updates. The decision on whether or not to deploy software updates are finalized by the providers themselves. Thus, if an update to a software services makes a client incompatible, the client has to either adapt their software or find another software service or even

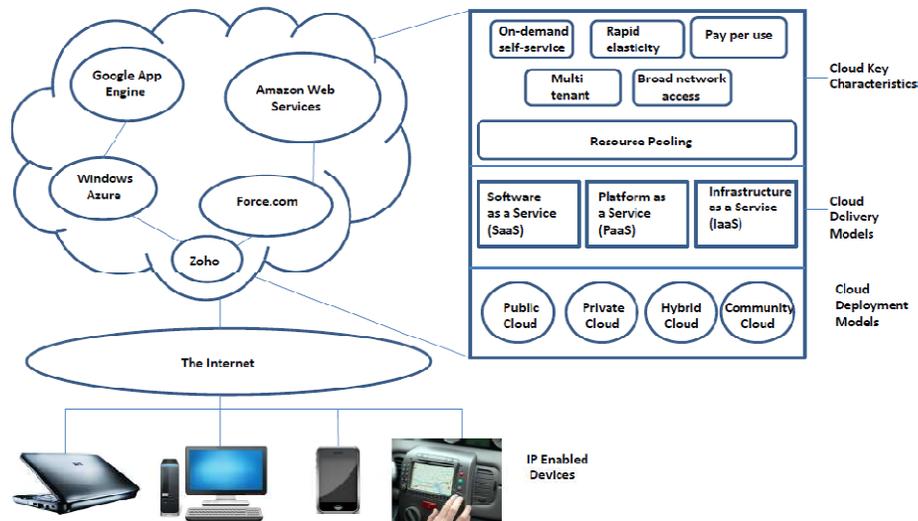


Fig 1. Cloud Computing High Level Infrastructure View

find another SaaS cloud. Applications with a Web-based interface accessed via Web Services and Web 2.0. Examples include Google Apps, SalesForce.com and social network applications such as FaceBook [5]. *Platform as a Service*: The PaaS category represents clouds that access to a range of compute, database and storage functions within a specified framework provided over the Internet. The benefit of PaaS clouds is clients are able to create their own required services and do not have to worry about provisioning and maintaining the hardware and software needed to run the services, infrastructure scaling, load balancing and so on. Examples include Force.com and Microsoft's Azure Platform [6].

Infrastructure as a Service: The IaaS category allows for the provisioning of hardware resources so cloud clients can create various configurations of computer systems from servers to complete clusters. On comparison to PaaS and SaaS, clients are able to create and use software as well as create and use an underlying software infrastructure to make the software possible. Grids of virtualized servers, storage & networks. Examples include Amazon's Elastic Compute Cloud and Simple Storage Service [7].

III. GOOGLE CLOUD COMPUTING SERVICES GOOGLE APP ENGINE

Google's providing both SaaS and PaaS solutions in cloud computing. Some of the example for SaaS solutions including Google Apps which including Gmail, Doc, etc., and PaaS includes Google App Engine [8].

In the Platform as a Service (PaaS) space Google is a key player. App Engine is a platform to create, store and run applications on Google's servers using development languages as java and python. App Engine includes tools for managing the data store, monitoring the site and its resource consumption, and debugging and logging. A user can serve the app from his own domain name (such as

<http://www.example.com/>) using Google Apps. Or, he can serve his app using a free name on the appspot.com domain. A user can share his application with the world, or limit access to members of organization. App Engine costs nothing to get started. All applications can use up to 1 GB of storage and enough CPU and bandwidth to support an efficient app serving around 5 million page views a month, absolutely free. Applications requiring more storage or bandwidth can purchase which is divided into five buckets: CPU time, bandwidth in, bandwidth out, storage, and outbound email. Google App Engine enables users to build a basic web application very quickly. Configuring and setting up an application is quick and easy. The Google App Engine Architecture provides a new approach without dealing with web servers and load balancers but instead deploying the applications on the Google App Engine cloud by providing instance access and scalability which is showing in the figure 2. The Google App Engine Software Development Kit (SDK) provides Java and Python programming languages. The languages have their own web server application that contains all Google App Engine services on a local computer. The web server also simulates a secure sandbox environment. The Google App Engine SDK has APIs and libraries including the tools to upload applications. The Architecture defines the structure of applications that run on the Google App Engine [9].

Python: The Google App Engine allows implementation of applications using Python programming language and running them on its interpreter. The Google App Engine provides rich APIs and tools for designing web applications, data modeling, managing, accessing apps data, support for mature libraries and frameworks like Django [10].

The main characteristics of Google App Engine are its DataStore, configuration file app.yaml and how it serves an application.

Bigtable: The DataStore is built on top of Bigtable which is built on top of Google File System. The DataStore is the only database Google App Engine supports for logging and storing data, including session data. It uses slightly different terminology inherited from the Bigtable. The Bigtable can be defined as a huge spreadsheet with unlimited number of columns and in the form of a string. The Bigtable is a tabular NoSQL database that is designed to reliably scale to petabytes of data and thousands of machines. It is a sparse, distributed, persistent, multi dimensional storage map. It is generally referred to as a "map" indexed with row key, column key and a time stamp. A "map" is an abstract data type composed of collection of keys, and a collection of values where each key is associated with one value. The Bigtable has a master server that coordinates the large segments of a logical table called "tablets". The tablets are split across a row with an optimal size of 200MB per each tablet for optimization purposes. The table contains rows and columns and each cell has a time stamp. The Google App Engine allows usage of the Bigtable in applications through the DataStore API [11].

The DataStore API: The DataStore is responsible for the scalability of the Google App Engine applications. The structure of the applications enables them to distribute the requests across the servers which should be compromised

with relational databases. Unlike any relational database the Google App Engine DataStore can create an infinite number of rows and columns that scales by adding servers to the clusters. In the DataStore, tables are called "models" and are represented in classes. Records are called "entities" and are instances of the model; columns are called "properties" and are attributes of models or entities. To access the DataStore we have to define a model class with some properties, then create entities and store them in database. Later queries can be run to retrieve the entities. The model class can be created by sub classing db.model. The Google App Engine provides a variety of property types from strings and integers to Boolean, date/time objects, list objects, phone numbers, email addresses, geographic points like latitude, longitude etc. The Google App Engine allows queries to be made using Bigtable as a database from its services using the Google App Engine Query Language (GQL) or Java Data Objects Query Language (JDOQL). All the data is being stored in the cloud which could be at any location on Google's servers. If data is to be stored on an external data base which is locally installed on our machine Google imposes strict constraints due to security issues which can be a potential problem if organizations put their secure data on servers located in remote locations. The Fig. 3 shows over all DataStore Architecture [12].

Configuration File "app.yaml" : The app.yaml file is a platform neutral and human readable file for representing data. It is created as an alternative to XML to represent structures in programming languages like lists and dictionaries. "Key: value" syntax represents items in a dictionary and "-" represents elements in a list. The file represents a dictionary with 5 key elements. They are application, version, runtime, api_version and handlers. The structure of the app.yaml file is shown in Figure 4. The first key is application, it can be any name when run on a local server. But if it is uploaded to the Google's server, then the key application value must be the Application ID value. The second key is "version" which is used to specify version number of application. Google uses "MAJOR.MINOR" format to represent application numbers. MAJOR version is the number user sets and MINOR version is nth upload of that version. The Google App Engine saves last upload for every MAJOR version, and one among them can be chosen as the current one. For the third and fourth keys runtime and api_version are specified as Python. Newer versions of API will be available in future. Handlers specify mapping of URL patterns. Handlers are different key values which can be a static file, script file or a static directory.

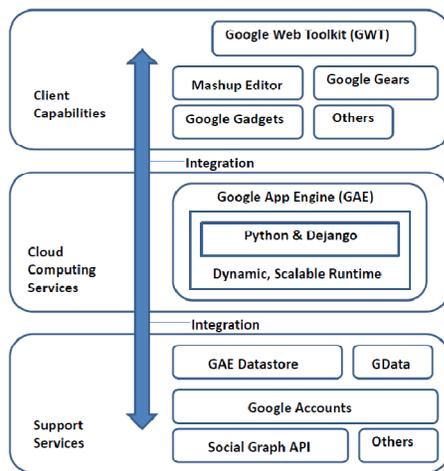


Fig. 2. Architecture of Google App Engine

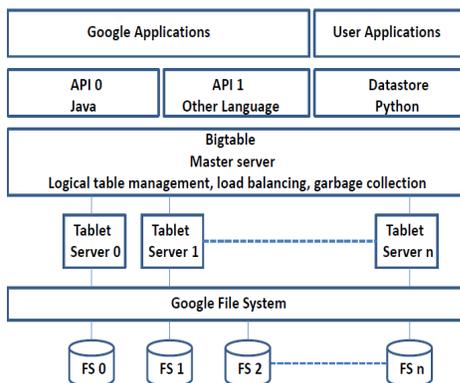


Fig. 3. Google DataStore Architecture

```

application: application ID registered with App Engine
version: 1
runtime: python
api_version: 1
handlers:
  - url: /*
    script: name of the .py file
    
```

Fig. 4. App.yaml File

How the App Engine serves applications. Each application has an `app.yaml` file which tells how to handle URL requests. Google App Engine provides a simple framework called `webapp` that helps to organize code. When a web browser sends a request to the Google's cloud it chooses a server near the users location, instantiates the application if it is not running and processes the users request. Therefore the cloud meets the demands by creating the instances when required and deletes them when they are not used.

Java: The Google App Engine provides tools and APIs required for the development of web applications that run on the Google App Engine Java run time. The application interacts with the environment using servlets and web technologies like Java Server Pages (JSPs) which can be developed using Java6. The GAE environment uses Java SE Runtime JRE platform 6 and libraries which the applications can access using APIs. Java SDK has implementations for Java Data Objects (JDO) and Java Persistence (JPA) interfaces. To exchange email messages with Google App Engine, it provides the Google App Engine mail service through the Java Mail API. Support for other languages like JavaScript, Ruby, or Scala is also provided by Google App Engine with the use of JVM compatible compilers and interpreters. When Google App Engine gets a web request that corresponds to the URL mentioned in the applications deployment descriptor (i.e., `web.xml` file in `WEB-INF` directory) it invokes a servlet corresponding to that request and uses Java Servlets API to provide requested data and accepts response data.

Google App Engine makes it easy to build an application that runs reliably, even under heavy load and with large amounts of data. App Engine includes the below features:

- Dynamic web serving, with full support for common web technologies
- Persistent storage with queries, sorting and transactions
- Automatic scaling and load balancing
- APIs for authenticating users and sending email using Google Accounts
- Scheduled tasks for triggering events at specified times and regular intervals

IV. AMAZON CLOUD COMPUTING SERVICES: AMAZON WEB SERVICES

Amazon's cloud computing known as Amazon Web Services (AWS) is a pioneer in cloud computing. AWS is a full-featured cloud platform with web services which span the IT infrastructure for example, servers, storage, databases, messaging, load balancing, content distribution, and distributed computing. The Amazon Web Services benefits of low costs, elasticity, and reliability can help organizations of all sizes – from the smallest startup to the largest enterprise [13].

The Architecture of the Amazon web service is mainly composed of four components namely Amazon Elastic Compute Cloud (EC2), Amazon Simple Storage Service (S3),

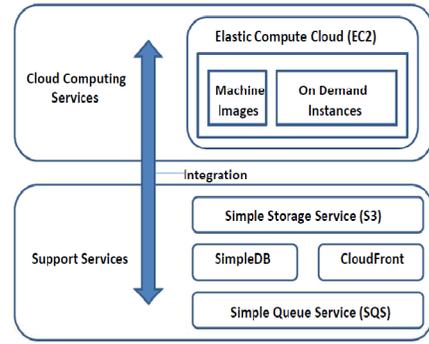


Fig 5. Architecture of Amazon Web Services

Amazon SimpleDB and Amazon Simple Queue Service (SQS). Some other value added service components also available that perform plugins enhancing attribute and functions.

A. Amazon Elastic Compute Cloud (EC2)

This service makes it simple to create, launch, and provision virtual instances. There are five different types of instances according to the usage of CPU, Memory, Hard Disk and IO Performance. An application needing a significant amount of RAM or CPU can rent more expensive but more powerful instances, while a network-bound application like a web-server can use cheaper and less powerful instances. To use Amazon EC2, a user need to create an Amazon Machine Image (AMI) containing users applications, libraries, data and associated configuration settings or a user can choose preconfigured template images which are already provided from Amazon, to start the instance immediately, after that upload the AMI into the amazon simple storage service S3. After starting an instance a user can configure security and network access of it. Also add more volumes attach to an instance or associate it with an elastic static IP address to communicate to the world. After running, a user can terminate instances and pay only for the resources that actually consume, like instance hours or data transfer [14].

Amazon Machine Image (AMI) is an encrypted machine image that contains all information necessary to boot instances of the software. For example, an AMI might contain Linux, Apache, and the web site or it might contain Linux, Hadoop, and a custom application. AMIs are stored in Amazon Simple Storage Service (S3). Amazon has multiple data centers in separate geographical locations. A user can launch the instance in different regions. One region (USA or Europe) has different availability zones within it. The advantage of launching instance in separate regions are application to be closer to specific customers or to meet legal or other requirements and also user can protect the applications from the failure of a single location. If user do not specify an availability zone when launching an instance, Amazon will automatically choose one based on the current system health and capacity [15].

B. Amazon Simple Storage Service (S3)

Amazon S3 is storage for the Internet which is a scalable, high-speed, low-cost Web-based service designed for online

backup and archiving of data and application programs. The S3 allows uploading, storage and downloading of practically any file or object up to five gigabytes in size. Subscriber data is stored on redundant servers in multiple data centers. The S3 employs a simple Web-based interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web and uses encryption for the purpose of user authentication.

S3 simply stores "objects" or files, in "buckets" (folders). Since there are no directories in S3, each bucket is given a unique identifier. User can also have multiple buckets under one account.

Many customers serve static files such as images or video directly from S3 instead of having them stored on a local disk. This gives them virtually infinite storage capacity for their files without purchasing any hardware [16].

C. Amazon SimpleDB

Amazon SimpleDB is a highly available, scalable, and flexible non-relational data store that enables to store and query data items using web services requests. It requires no schema, automatically indexing and providing a simple API for storage and access. Data is accessed by http through REST and SOAP protocols. SimpleDB consists of multiple domains and each domain stores a set of records. Each record has a unique key and a set of attributes, which may or may not be present in every record. Within each domain, the records are automatically indexed by every attribute. The main operations are to read/write a record by key or to search for a set of records by one or more attributes.

D. Amazon Simple Queue Service (SQS)

It provides highly scalable and available message passing channel between cloud computing components. For example if the application is divided into 2 processes running in two Amazon EC2 instances and the second process needs a result from the first process, the Amazon SQS can receive the message from the first, store it until the second ready to receive and forward the message to the second reliably. Using this queue offers higher scalability and availability then passing the message directly [17]. This is because the first does not have to wait for the second to retrieve the message and the second can still receive the message through the first may be unavailable.

V. MICROSOFT CLOUD COMPUTING SERVICES: WINDOWS AZURE

Windows Azure is Microsoft's Platform as a Service cloud computing solution. It consists of five main parts: Compute, Storage, Content Delivery Network (CDN), SQL Azure and the base fabric to hold everything together across a heterogeneous network. Fig. 6 shows a high level architecture of Azure's structure.

A. Compute Service

The compute part of the Windows Azure platform is responsible for providing CPU cycles for executing applications. Applications are hosted inside virtualized

environments to prevent any physical dependencies on the underlying operating system and hardware. Loose coupling of applications is accomplished through virtualized resources, which include local files, persistent storage (structured and unstructured), and diagnostic and instrumentation resources. The hosting environment is implemented as a virtual machine, thus any application failures won't impact other applications running on the same physical hardware [18].

Applications are deployed into the Windows Azure platform as packages of roles and associated executable code and resources. An Azure role describes the characteristics of the hosting environment declaratively. When a deployed application is activated, the Azure provisioning environment parses the service model, selects a preconfigured virtual machine (VM) image based on the role type, copies the application bits to the VM, boots the machine and starts the necessary application services. Web Roles include Internet Information Services (IIS) and can accept HTTP and HTTPS Requests. Worker Roles aren't configured with IIS and are primarily for background processing, such as queuing, monitoring, or ticket-system scenarios.

B. Storage Service

Windows Azure Storage allows to store data for any length of time and to store any amount of data.

Blobs: These are used for binary data and are like files on our hard drive. There are two types – block and page. Block blobs are used for streaming, like when user want to start watching a movie before it even completes the download. It can store files up to 200GB at a pop. Page blobs are used when user need a LOT of storage (up to a terabyte) and access a "page" directly, with an address [19].

Tables: To allow applications to work with data in a more fine-grained way, Windows Azure storage provides tables. These aren't relational tables. A table has no defined schema; instead, properties can have various types, such as int, string, Bool, or DateTime. A single table can be quite large, with billions of entities holding terabytes of data, and Windows Azure storage can partition it across many servers if necessary to improve performance. *Queues:* This storage is used to transfer messages between blocks of code.

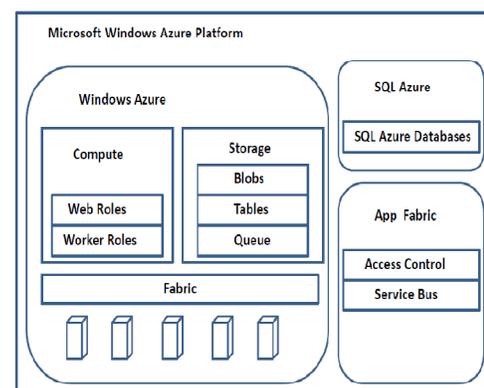


Fig. 6. High Level Architecture of Microsoft Windows Azure

C. SQL Azure

SQL Azure is a cloud-based service from Microsoft offering data storage capabilities similar to Amazon S3. Unlike similar cloud-based databases, SQL Azure allows relational queries to be made against stored data, which can either be structured or semi-structured, or even unstructured documents, user can connect to SQL Azure in a variety of ways, such as ADO.NET, PHP, and Open Database Connectivity (ODBC). SQL Azure features querying data, search, data analysis and data synchronization. SQL Azure uses Microsoft SQL Server as a backend, but it exposes only a subset of the data types — including string, numeric, date and Boolean. It uses an XML-based format for data transfer. Like Microsoft SQL Server, SQL Azure uses T-SQL as the query language and Tabular Data Stream (TDS) as the protocol to access the service over internet. It does not provide a REST-based API to access the service over HTTP. SQL Azure is available in three database sizes: 1 GB, 10 GB, and 50 GB. Your bill is based on the size of the database, not on the amount of information you actually store [20].

D. Content Delivery Network (CDN)

The CDN stores copies of a blob at sites closer to the clients that use it. The Windows Azure CDN actually has many more global caching locations than it shows, but the concept is correct. The first time a particular blob is accessed by a user, the CDN stores a copy of that blob at a location that's geographically close to that user. The next time this blob is accessed, its contents will be delivered from the cache rather than from the more remote original. For example, suppose Windows Azure is used to provide videos of a day's sporting events to a far-flung audience. The first user who accesses a particular video won't get the benefit of the CDN, since that blob isn't yet cached in a closer location. All other users in the same geography will see better performance, since using the cached copy lets the video load more quickly [21].

App Fabric: Windows Azure platform AppFabric (formerly called “.NET Services”) provides common application infrastructure services for distributed applications. The key components are:

Access Control – provides authentication and authorization services through rules and claims. It's a standards-based service that supports multiple credentials and relying parties. It enables federated identity and supports Active Directory, as well as other identity infrastructures.

Service Bus - provides secure connectivity options for service endpoints that would otherwise be difficult or impossible to reach. It supports various communication patterns such as relayed, buffered, bidirectional, publish-subscribe, multicast, streaming and direct-connect. Service Bus provides infrastructure for large-scale event distribution, naming, and service publishing. It helps address the challenges of firewalls, NATs, dynamic IP, and disparate identity systems. Service Bus can provide a service with a stable Uniform Resource Identifier (URI) that you can be accessed by any authorized client application. Service Bus supports REST and HTTP Access from non-.NET platforms and it supports standard protocols and extends similar

standard bindings for Windows Communication Foundation (WCF).

E. Windows Azure Fabric Controller

The Azure Fabric Controller (FC) is the part of the Windows Azure platform that monitors and manages servers and coordinates resources for software applications. The Azure Fabric Controller functions as the kernel of the Azure operating system. It provisions, stores, delivers, monitors and commands the virtual machines (VMs) and physical servers that make up Azure. The generic term fabric is a synonym for framework. Microsoft uses it in a proprietary manner to describe the servers, high-speed connections, load balancers and switches that make up the Azure cloud computing platform. The term fabric controller can generally be applied to any component that manages complex connections, but such components are often called by proprietary names.

VI. SALESFORCE.COM CLOUD COMPUTING SERVICES: FORCE.COM

In the PaaS area, Salesforce.com provides a development platform called force.com. Like other cloud computing development platforms, force.com allows users to access a application development and execution platform from a browser. This allows organizations to develop applications/products without making large scale investments in procuring software and hardware. Force.com can be divided into four major services namely Database, business logic, user interface, and integration.

The force.com platform architecture includes a database, a workflow engine, and user interface design tools. The platform includes an Eclipse-based IDE and a proprietary programming language called Apex. Apex has Java like syntax. The database is a relational database. It is not possible to run any Java or .Net programs on the force.com platform - developers have to use Apex to build applications. Force.com includes a tool called builder for building web applications quickly. Builder provides a user interface to create objects, fields within objects, and relationship between fields. Once a user creates these objects, the builder automatically creates a web interface with Create, update and delete operations. Using the builder allows developers to build simple to moderate applications without writing any significant amount of code and in a reasonably small amount of time. The platform also provides a rich reporting environment for plotting bar graphs and pie charts [22].

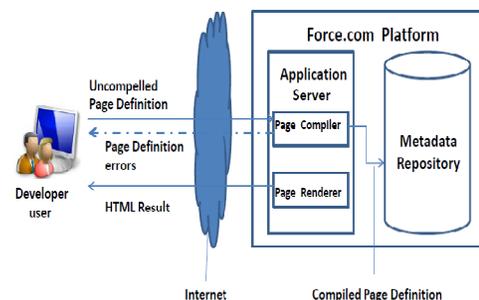


Fig. 7. Architecture of Force.com

TABLE 1: COMPARISON BETWEEN VARIOUS CLOUD COMPUTING PLATFORMS

	Google App Engine	Amazon Web Services	Windows Azure	Force.com
Cloud Services	PaaS	PaaS, IaaS	PaaS, IaaS	SaaS
Platforms Supported	Linux operating Systems Windows Server 2008	Linux Operating Systems Open Solaris Windows Server 2003	Windows Server 2003 Windows Server 2008	Linux Operating Systems Windows Server 2008
Virtualization Platform	Application Container	OS level running on a Xen Hypervisor	Xen Hypervisor	Not Available
Storage	BigTable and MegaStore	Amazon Simple Storage Service and SimpleDB	SQL Server Data Services (SSDS)	Database deals in terms of relationship fields
Auto Scaling	Automatic scaling which is transparent to users	Automatically changing the number of instances based on parameters that users specify	Automatic scaling based on application roles and a configuration file specified by users	real time scalability
Control Interface	API (Application Programming Interface)	API (Application Programming Interface) Command Line	API (Application Programming Interface), Command Line	API (Application Programming Interface) , Web Based Application/ control panel
Languages Supported	MapRduce programming framework that supports, Java Python	Java, PHP, Python Ruby, WinDev, Amazon Machine Image (AMI)	Microsoft .NET	Java PHP Ruby
Load Balancing	Automatic scaling and load balancing	Round-robin load balancing. Service will allow users to balance incoming requests and traffic across multiple EC2 instances.	Built-in hardware load balancing	Load balancing among tenants
Fault tolerance	App Engine Corn Service	System should automatically alert, failover and re-sync back to the "last known state" as if nothing had failed.	Containers are used for load balancing and availability	Self management and self tuning
Interoperability	Interoperability between platforms of different vendors and programming languages	Supports horizontal Interoperability e.g Interoperability among EC2 etc.	Interoperable platform can be used to build new applications to run from the cloud or enhance existing applications with cloud-based capabilities	Application level integration between different clouds
Service Level Agreements (SLA) Availability	100% Uptime	Amazon S3 - 99.9% EC2 - 99.95 % Uptime	99.9 % uptime	99.9+ percent uptime
Support response time	Not available	Gold Support: 1 hour High-4business hour Normal-1 business day	Not available	Basic Support -2 days Premier Support- 2 hours Premier Support with Administration – 2 hours
Choice of data hosting location Data after termination	USA-North Virginia USA-North California Europe- Ireland	Not Available	USA- Anywhere USA- North West USA-South West.	Not Available
Data after termination	Google will not take any action for 90 days after the effective date of termination	Amazon will not take any action for a period of 30 days after the effective date of termination	For retention period user must contact Microsoft before termination of the service.	After the request made by user within 30 days after the effective date of termination.
Data Security	Google Secure Data Connector (Uses RSA/128-bit or higher AES CBC/SHA and TLS-based server authentication)	Type-II (SAS70 Type II) certification, firewall, X.509 certificate, SSL-protected API, Access Control List	Filtering routers Firewalls Physical security	SysTrust SAS 70 Type II Users and security, programmatic and platform security framework

VII. IDENTIFICATION OF A COMPANY'S SUITABILITY FOR ADOPTING OF CLOUD SERVICE

A company should consider given below key characteristics of the resources before hopping onto the cloud Cloud Computing.

Sensitivity of the Data: It is one of the major criteria for whether migrate a application to cloud or not. There are mainly 4 types of data classification namely Extremely Sensitive, (Example: Government federal and intelligence agencies like CIA, FBI), Very Sensitive (Banking and Financial related data), Sensitive (Hospital Database), Not Sensitive (music, videos, pictures, games, news, articles).

Criticality of the Application: Highly criticality of the Application requires strength of resources, platforms and security, for example for a manufacture company, the main production support applications could be CRM (Customer Relationship management) and SCM (Supply chain management) highly critical. So these types of applications do not fit for cloud.

Utilization Pattern of the Resources: A large number of much underutilized server resources leads to a large amount of wastage and its better to makes it suitable for using cloud services, whereas perennially well utilized servers would not be economical if taken to the clouds.

Size of the IT infrastructure: The larger the volume of the resources, the lesser is the per capita cost of the operating these resources. It then becomes much cheaper and more beneficial in the long run than using the cloud services. There are many factors taken into account while determine the size of the IT resources of a company like size of the data center, annual revenue from IT, Employee strength and the number of countries across which the company spreads its operation.

VIII. CRITERIA FOR SELECTING A CLOUD SERVICE

Like any other service, a user always knows what he is paying for and what measurements exist to show that services actually receiving by the service provider.

SLA (Service Level Agreement): SLA offered by the vendor, some examples are Availability of the services, uptime, downtime, process of switching from one provider and data archiving etc. should fit into the organizational policies.

Selection of Vendor: Vendor should understand the needs and business requirements of the customer and offer the best possible solution at an affordable price is very important.

Hardware Infrastructure: Careful evaluation of the infrastructure is needed when using cloud services like Internet connection, routers, firewall, Wi-Fi points, local backup server and printers are some constituents of infrastructure which will still be needed.

Performance related Issues: Issues like high network latency, congestion, and slow network speed can become a routine problem and affect the overall work and timeline of deliverables. A solution to this problem is utilizing the

geographically located content delivery networks setup by cloud providers; this would decrease the data travelling time and prevent congestion thereby increasing the performance.

Data Security: It is necessary to evaluate the providers security by finding out whether the service provider is a regular target of hacking attempts or how often have his systems been compromised, confidentiality of information, ensuring that information flow is not shared.

Data Security: It is necessary to evaluate the providers security by finding out whether the service provider is a regular target of hacking attempts or how often have his systems been compromised, confidentiality of information, ensuring that information flow is not shared.

Technical Integration: Many enterprise applications cannot be completely migrated to cloud either due to cloud restrictions or due to data dependencies with other on-premise applications. So it is important to decide how and which applications need to be moved to cloud.

Application Migrating to Another Cloud: There can be a situation where an application is migrated to cloud but later the application needs to be migrated to other cloud, there is a risk here as migration costs could be significant and some cloud service providers require a long-term contract to avoid setup fees. So, the cost and benefits of moving an application to cloud should be analyzed properly before actual application migration.

IX. CONCLUSION

Cloud computing has dramatically changed how business applications are built and run. At its core, cloud computing eliminates the costs and complexity of evaluating, buying, configuring, and managing all the hardware and software needed for enterprise applications. Instead, these applications are delivered as a service over the Internet. Cloud computing is a powerful new abstraction for large scale data processing systems which is scalable, reliable and available and also it needs to be extended to support negotiation of QoS based on Service Level Agreement (SLA's). Cloud computing is particularly valuable to small and medium businesses, where effective and affordable IT tools are critical to helping them become more productive without spending lots of money on in-house resources and technical equipment. Also it is a new emerging architecture needed to expand the Internet to become the computing platform of the future.

REFERENCES

- [1] Buyya, R., Yeo, C. S., Venugopal, S., Broberg J. and Brandic, I. 2009, "Cloud computing and emerging IT platforms: vision, hype and reality for delivering computing as the 5th utility", FGCS, 25(6), pp. 599-616.
- [2] M. Armhurst et al., "Above the Clouds: A Berkeley View of Cloud Computing" Comm. ACM, vol. 53, no. 4, Apr. 2010, pp. 50-58.
- [3] Y. Shen, K. Li and L. T. Yang, "Advanced Topics in Cloud Computing", Journal of Network and Computer Applications, vol. 12, pp. 301-310, 2010.

- [4] Grossman, R.L., Gu, Y., Sabala M. and Zhang W. 2009, "Compute and storage clouds using wide area high performance networks", *Future Generation Computer Systems*, 25 (2), pp. 179-183.
- [5] Choudhary .V, "Software as a Service: Implications for Investment in Software Development", *Proceedings of 40th Hawaii International Conference on System Science*, 2007
- [6] S. Subashini, V. Kavitha, (2011) "A survey on security issues in service delivery models of cloud computing", *Journal of Network and Computer Applications* 34 (2011) 1–11.
- [7] Sean Carlin, Kevin Curran,(2011) "Cloud Computing Security", *International Journal of Ambient Computing and Intelligence*, Vol. 3, No. 1, pp: 38-46, April-June 2011, ISSN: 1941-6237, IGI Publishing.
- [8] Google App Engine documentation link. <http://code.google.com/appengine/docs/whatisgoogleappengine.html>
- [9] Rao Mikkilineni, Vijay Sarathy "Cloud Computing and Lessons from the Past", *Proceedings of IEEE WETICE 2009, First International Workshop on Collaboration & Cloud Computing*, June 2009.
- [10] Muzafar Ahmad Bhat et al, *Cloud Computing: A solution to Geographical Information Systems (GIS)*, *International Journal on Computer Science and Engineering (IJCSE)*, ISSN : 0975-3397 Vol. 3 No. 2, Feb 2011.
- [11] Lijun Mei, W.K.Chan and T.H.T se, —A Tale of Clouds: Paradigm comparisons and some thoughts on research issues, 2008 IEEE Asia-Pacific Services Computing Conference.
- [12] Liladhar R. Rewatkar, Ujwal A. Lanjewar, *Implementation of Cloud Computing on Web Application*, *International Journal of Computer Applications (0975 – 8887) Volume 2, No.8, June 2010.*
- [13] Amazon Web Services documentation link. <http://aws.amazon.com/documentation/>
- [14] Amazon elastic compute cloud (EC2). [http://www.amazon.com/ec2/\(10.06.11\)](http://www.amazon.com/ec2/(10.06.11)).
- [15] Cong Wang, Qian Wang and Kui Ren. —Ensuring Data Storage Security in Cloud computing|| 978-1-4244-3876-1/2009 IEEE.
- [16] Amazon simple storage service (S3). [http://www.amazon.com/s3/\(01.05.11\)](http://www.amazon.com/s3/(01.05.11)).
- [17] Rabi Prasad Padhy, Manas Ranjan Patra, Suresh Chandra Satapathy, "RDBMS to NoSQL: Reviewing Some Next-Generation Non-Relational Database's", *International Journal of Advanced Engineering Science and Technologies*, Vol. 11, No.1, pp: 15-30.
- [18] Windows Azure Platform documentation link. <http://msdn.microsoft.com/en-us/library/dd163896.aspx>
- [19] S. Subashini and V. Kavitha, "A Survey on Security Issues in Service Delivery Models of Cloud Computing", *Journal of Network and Computer Applications*, vol. 34, no. 1, pp.1-11, 2010.
- [20] E.Orakwue, "Private Cloud: Secure managed services", *Information Security Journal: A Global Perspective*, vol. 19, no. 6, pp. 295-298, 2010.
- [21] K. Mukherjee, G.Sahoo, *Development of Mathematical Model for Market-Oriented Cloud Computing*, *International*

Journal of Computer Applications (0975 –8887) Volume 9– No.11, November 2010.

[22] Force.com documentation

[23] <http://wiki.developerforce.com/index.php/Documentation>

ABOUT THE AUTHORS



ITIL Certified.

Rabi Prasad Padhy is currently working as a Senior Software Engineer - Oracle India Private Ltd. Bangalore. He has achieved his MCA degree from Berhampur University. He carries 8 years of extensive IT Experience with MNC's like EDS, Dell, IBM and Oracle. His area of interests include IT Infrastructure Optimization, Virtualization, Enterprise Grid Computing, Cloud Computing and Cloud databases. He has published several research papers in international journals. He is a certified professional for Oracle, Microsoft SQL Server database and also



Dr. Manas Ranjan Patra holds a Ph.D. Degree in Computer Science from the Central University of Hyderabad, India. Currently he is an Associate Professor in the Post Graduate Department of Computer Science, Berhampur University, India. He has about 24 years of experience in teaching and research in different areas of Computer Science. He had visiting assignment to International Institute for Software Technology, Macao as a United Nations Fellow and for sometime worked as assistant professor in the Institute for Development and Research in Banking Technology, Hyderabad. He has about 90 publications to his credit. His research interests include Service Oriented Computing, Software Engineering, Applications of Data mining and E-Governance. He has presented papers, chaired technical sessions and served in the technical committees of many International conferences.



Dr. Suresh Chandra Satapathy is a Professor and Head of the Dept of Computer Science and Engg. in Anil Nerrukonda Institute of Technology and Sciences (ANITS), Vishakapatnam, India. He has published many IEEE conference papers on Data clustering using PSO, GA etc. His areas of interest include Data mining, machine learning, Swarm Intelligence etc. He is a Senior Member of IEEE and Computer Society of India.