On Web Services Based Cloud Interoperability

Devi Prasad Bhukya[1], Reeta Sony A.L[2], Gautam Muduganti
[1]Council of Scientific & Industrial Research, New Delhi, India
[2]National Law University, New Delhi, India

Abstract

Cloud Computing is a paradigm shift in the field of Computing. It is moving at an incredible fast pace and one of the fastest evolving domains of computer science today. It consist set of technology and service models that concentrates on the internet base use and delivery of IT applications, processing capability, storage and memory space. There is a shift from the traditional in-house servers and applications to the next generation of cloud computing applications. With many of the computer giants like Google, Microsoft, etc. entering into the cloud computing arena, there will be thousands of applications running on the cloud. There are several cloud environments available in the market today which support a huge consumer-base. Eventually this will lead to a multitude of standards, technologies and products being provided on the cloud. Consumers will need certain degrees of flexibility to use the cloud application/services of their choice and at the same time will need these applications/services to communicate with each other. This paper emphasizes cloud computing and provides a solution to achieve Interoperability, which is in the form of Web Services. The paper will also provide a Live Case Study where interoperability comes into play – Connecting Google App Engine and Microsoft Windows Azure Platform, two of the leading Cloud Platforms available today. GAE and WAP are two Cloud Frameworks which have very little in common, making interoperability an absolute necessary.

Keywords – Cloud Computing, Grid Computing, Cluster Computing, web service

I. INTRODUCTION

Cloud Computing (CC) [1][2] is one of the fastest evolving domains of computer science in today's world. There is a lot of research being pursued in this sector and lot of scope for ground-breaking inventions in this field. There are several consumer-driven applications being developed under this domain using various cloud environments. Many of the big players of the industry like Google [3], Microsoft [4], IBM [5], Oracle [6], Amazon [7] etc. have spread their wings into CC, clearly emphasizing that this is the next big thing. But, with many companies, many standards, many technologies and so many products, the most difficult task is going to be the integration of the various applications running in these cloud environments. However, interoperability will be the vital key for the sustenance of CC in years ahead. The outline of this paper is as follows: Section 2 explains about overview of CC; Section 3 discusses cloud computing deployment models; Section 4 discuss about why interoperability; Section 5 explains about Web Services; Section 6 discusses about the Web Services for Interoperability; Section 7 discuss about Connecting GAE and WAP – A Case Study, Section 8 explains about selecting the appropriate WebService flavor for Interoperability; Section 9 discuss about Case Study Analysis and Section 10 presents the conclusions

II. OVERVIEW OF CLOUD COMPUTING

CC is a paradigm in the field of computing. It is a style of computing where resources are offered as a metered service. These resources are generally dynamically scalable and virtualized. CC builds on existing paradigms like Distributed Systems, Grid Computing, Cluster Computing [17], Virtualization [15], etc. In other words – "CC is the provision of dynamically scalable and often virtualized resources as a service over the Internet on a utility basis." CC services often provide common business applications online that are accessed from a web browser, while the software and data are stored on the servers. The term cloud is used as a metaphor for the Internet, based on how the Internet is depicted in computer network diagrams and is an abstraction of the underlying infrastructure it conceals. This definition states that clouds have seven essential characteristics –

- On-demand Self Service
- Broad Network Access
- Cost effective
- Reliability
- Resource Pooling
- Rapid Elasticity
- Measured Service

A DELIVERY MECHANISM

CC is mainly built on delivery mechanisms. These mechanisms define the service being provided by an application running in a CC Environment. Commonly these mechanisms form "as a Service (aaS)" [8] models. There are hundreds of aaS models like Globalization as a Service (GaaS), Communication as a Service (CaaS), etc. However there are four important delivery mechanisms typically used –

A1 Infrastructure as a Service (IaaS)

Infrastructure as a Service is the delivery of computer infrastructure as a service. The primary philosophy of this service is to buy servers, data centers, etc. as an outsourced service rather than traditionally hardware. Examples of applications following Infrastructure as a

Service are – Amazon Elastic Compute Cloud (Amazon EC2).

A2 Platform as a Service (PaaS)

Platform as a Service is a framework definition for the deployment of applications over the Internet. This delivery mechanism concentrates on building up web servers, IDEs, etc. for the user to host their applications. This mechanism is closely tied with the SaaS mechanism. Examples of applications following Platform as a Service are – Google App Engine, Microsoft Windows Azure Platform.

A3 Software as a Service (SaaS)

Software as a Service is a model of software deployment where an application is hosted as a service provided to customers across the Internet. SaaS eliminates the need to install and run the application on the customer's own computer. Applications built on this delivery mechanism are customer centric and focus on providing a particular function to the users. Examples of applications following Software as a Service are – Google Docs, Email, Salesforce, etc.,

A4 Data as a Service (DaaS)

Data as a Service provides customers with access and analytics around a proprietary set of aggregated data. Fundamentally DaaS aims at building up a repository of information and disturbing it across a multitude of users. Examples of applications following Data as a Service are – Salary.com

III DEPLOYMENT MODELS

A Private Cloud

The cloud infrastructure is operated solely for an organization. It may be managed by the organization or a third party and may exist on premise or off premise. Eg, Amazon VPC (Virtual Private Cloud), VMware Cloud Infrastructure Suite, and Microsoft ECI data center, etc.

B Public cloud

The cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services. Eg Google App Engine, Microsoft Windows Azure, IBM Smart Cloud and Amazon EC2,etc.

C Community cloud

The cloud infrastructure is shared by several organizations and supports a specific community that has shared concerns. It may be managed by the organizations or a third party and may exist on premise or off premise. Eg, Microsoft Government Community Cloud.

D Hybrid cloud

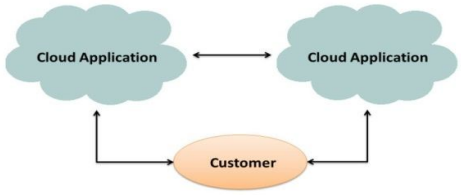
The cloud infrastructure is a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability.

IV WHY INTEROPERABILITY?

The importance of interoperability is closely linked to the importance of CC for a customer. Understanding "Why Interoperability" would actually go down to understanding "Why CC". From a customer's perspective, CC is a cost-effective solution. CC offers flexibility in usage and the expenditure is only as per the use. There are no troubles of setting up hardware infrastructure like servers and storage [16] for the customers. For a vendor, CC offers a high degree of customizability and at the same time allows vendors to reach more and more customers with low cost of delivery. So CC is a viable solution for both customers and vendors. This will eventually lead to thousands of applications on the Cloud.

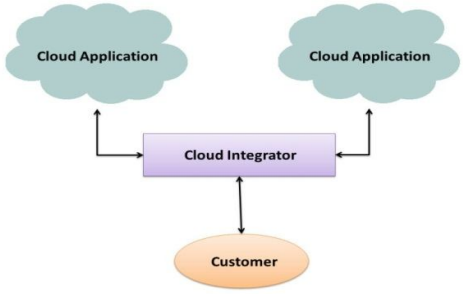
A Interoperability as a Cloud Provider

There are several applications which run on cloud platforms today. This is a typically a "Software as a Service"[9] using a "Platform as a Service" model, since these applications provide a service to several other customers. The application developers can be termed as Cloud Providers because of the SaaS model and the Platform providers are also Cloud Providers because of the PaaS model. There are several Cloud Environments available today and each one of them offers their own services. In a few years from now, there will a lot of legacy applications for Enterprise running on these proprietary clouds. A migration to another cloud environment will mean re-writing these applications which will be colossal hit on the organizations (Cloud Providers) economy. An economically sound plan would be building the new application at the new Cloud Environment and at the same allowing these new applications to communicate to the legacy applications.



B Interoperability as a Customer

There are several applications built with similar features. Examples of these applications include Sales Management Systems, Pay-roll applications, etc. With these many choices, customers will have the flexibility of choosing what they want depending on the pricing of the application, its efficiency, the various optimization techniques it uses, etc. Customers would like these applications to communicate which each other to establish a portal using them. These two scenarios are exactly what interoperability aims at, making it a vital component of CC.



V WEB SERVICES

Web Services [10] is a standardized way of integrating Web-based applications using the XML, SOAP, WSDL and UDDI open standards over an Internet protocol and its hottest buzz-word in computing today. Prior to Web services, interoperability and integration were major hurdles in application development. Limited integration and interoperability took place amidst numerous technologies, vendors, obstacles and formats that prevented sharing of data. Then Web Services technology came along and changed all that. Web Services is a technology for transmitting data over the Internet and allowing programmatic access to that data using standard Internet protocols. It is this programmatic interface that allows two applications to be integrated.

The best part of Web Services is that it allows a developer to include various functionalities into a program without the need of "reinventing the wheel" and without needing to know anything about the business logic or complexity of the Web Service being used. Web Services today are frequently just application programming interfaces (API) or web APIs that can be accessed over a network, such as the Internet, and executed on a remote system hosted the required services. Web Services implementation typically consists of two stages – Service Publish and Service Consume. In addition, for Big Web Services there is a third stage – Service Discovery.

A Big (SOAP) Web Services

Big Web Services [11] are primarily based on the Simple Object Access Protocol (SOAP) protocol stack. SOAP messages form the primary crux of this model. On a conceptual level, Big Web Services use messages to exchange invocation request and response information between the service producer and service consumer. On a technology level, Big Web Services communicate using SOAP messages. SOAP is a protocol specification for exchanging structured information using Extensible Markup Language (XML). Service Discovery is through UDDI (Universal Description Discovery and Integration) via a Service Broker. The Service Producer registers his service with the Service Broker using UDDI Register and the Service Consumes gets a reference to the Web Service through UDDI Inquiry. The Service consumer then gets a description of the Web Service methods using the WSDL (Web Services Description Language). The Service Consumer consumes the service through a series of request and responses built as SOAP messages. Google seems to be consistent in implementing their web services using SOAP.

B RESTful Web Services

A major inspiration for RESTful Web Services [12] has been REST. This is clearly evident from the name itself. REST stands for "Representational State Transfer" and is primarily a style of software architecture for distributed hypermedia systems like the World Wide Web. Conforming to the REST constraints is termed as being RESTful. The Web is comprised of resources. A resource is any item of interest. Clients may access these resources through URLs (Uniform Resource Locators). Then a representation of these resources is returned. This representation places the client application in a state. REST is an architectural style and not standard, but uses several standards like HTTP (hypertext Transfer Protocol), URL, XML, MIME (Multipurpose Internet Mail Extensions), etc. In fact HTTP forms the very basis of REST as most of the resource handling in REST is mapped to the traditional HTTP methods like GET, POST, DELETE, etc. The latest generation of Web Services has been influenced by REST to a very large extent. RESTful Web Services are gaining a lot of momentum in the industry, especially with Internet companies. By using HTTP methods like PUT, GET and DELETE alongside POST, these are often better integrated with HTTP and web browsers than SOAP based services. They do not require XML messages or WSDL service-API definitions. All of Yahoo's web services use REST, including Flickr, del.ico.us, etc.

VI WEB SERVICES FOR INTEROPERABILITY

It is clearly evident that interoperability is a necessity in CC today. Web Services provide a flexible way of communication and exposing business logic components making them the center of attraction for cloud interoperability. The two variations (SOAP and REST)

greatly support the communication of cloud applications with each other and at the same time allow customers to integrate the cloud services to their applications. The SOAP protocol works using XML messages, so any programming language which can understand XML can be used as a Web Service Provider and a Web Service Consumer. Several languages we use in the development of our applications like – Java, C#, etc. understand XML making them viable languages for developing SOAP web services. SOAP Web Services are easy to consume. They have a rigid type-checking and adhere to a contract. They have several development tools to build and consume them. On the other hand, REST has an advantage over SOAP in terms of their exposure through HTTP methods. The language using these web services need not use any special technology like SOAP; they can access them through just pure HTTP requests. So, even programming languages which cannot XML like C++ can use it with the help of support frameworks. REST Web Services are lightweight. They are presented in human readable format. They are easy to build without the need of special frameworks and toolkits.

VII CONNECTING GAE AND WAP – A CASE STUDY

Google App Engine (GAE) and Windows Azure Platform (WAP) are two of the biggest cloud environments available in today's market. Connecting applications in these two cloud environments makes a perfect case study of cloud interoperability.

A Google App Engine

Google App Engine [13] is a CC technology from Google. It is a platform for developing and hosting web applications in Google-managed data centers. Primarily, Google App Engine lets users to run their web applications on Google infrastructure. It virtualizes applications across multiple servers and data centers. Google App Engine supports two primary languages – Python and Java. And by extension, other JVM languages like Groovy, JRuby, Scala and Clojure. Python web frameworks that run on Google App Engine include Django, CherryPy and web2py. For persistence requirements, Google App Engine provides a data-store along with an API for it. To access the data in it, a language with a SQL-like syntax called "GQL" has been unveiled.

B Windows Azure Platform

The Windows Azure Platform [14] is a cloud platform from Microsoft. It provides a wide range of services to be consumed from on-premises environments or the Internet. It is primarily an application platform in the cloud that allows applications to be hosted and run at Microsoft datacenters. It consists of a cloud operating system called Windows Azure that serves as a runtime for the applications and provides a set of services. The core programming languages for the Windows Azure Platform is through .NET (C# and Visual Basic), C++. Windows Azure Platform is also compatible with other programming languages like PHP, Ruby, Python and Java. To support persistence requirements, Windows Azure Platform is integrated with Microsoft SQL Azure, a cloud-based relational database. SQL Azure supports Transact-SQL (T-SQL).

VIII SELECTING THE APPROPRIATE WEB SERVICE FLAVOR FOR INTEROPERABILITY

Selecting the appropriate Web Service flavor for the cloud environment is a very important decision while building business components. This decision will vary from application to application but broadly depends on the following –

- The service consumers programming language
- Limitations of the target cloud environments
- Usage of advanced features like Security, Trust, Transactions, BPEL (Business Process Execution Language), etc.

Coming to the case study of GAE and WAP, we can narrow our decision to RESTful Web Services because of one strong reason – though GAE supports Java, it doesn't support JAX-WS, the component required for SOAP Web Services.

IX CASE STUDY ANALYSIS

As an example of Interoperability, consider a HR System running on the Windows Azure Platform. The employee details for the HR system can be stored using Azure Storage or SQL Azure. The HR system is built as a simple CRUD web application on the data-store. Consider the requirement of another web application on the Google App Engine – a Name Finder. The data-store needed by the name finder is exactly the same one which was used in the HR System. This is a typical case scenario for Cloud Interoperability – two Cloud Applications running on different Cloud environments need to communicate with one another. The easiest solution for this would be exposing the data-store of the HR System as a RESTful Web Service so that the Name Finder can access and use it.

X CONCLUSION

Interoperability is a key factor in the next generation CC applications. Various techniques are being researched to achieve cloud interoperability. In the present scenario, existing concepts like Web Services can be used effectively to obtain it. Typically every application built can be divided into three layers – the presentation; the business logic layer and the persistence layer.

Interoperability deals with the business logic and the persistence layers. Cloud applications should isolate the business logic and persistence layers from the presentation layers. Depending on the requirement, the business logic components along with the persistence layer can be exposed using SOAP or REST web services to allow other applications/users to connect to it.

REFERENCES

[1] Shuai Zhang, Shufen Zhang, Xuebin Chen, Xiuzhen Huo, "Cloud Computing Research and Development Trend", Second IEEE International Conference on Future Networks, 2010
[2] Rich Maggiani, solari communication. "cloud computing is changing how we communicate".
[3] Mike Burrows. The Chubby lock service for loosely-coupled distributed systems. 2006, http://labs.google.com/papers/chubby-osdi06.pdf
[4] Microsoft's Windows Azure Platform. http://www.microsoft.com/azure/ default.mspx
[5] IBM Research Division, Thomas J. Watson Research Center, 19 Skyline Drive, Hawthorne, New York 10532, USA
[6] Oracle Cloud ww.oracle.com/technetwork/topics/cloud/index.htm
[7] Sangho Yi, Member,Artur Andrzejak, and Derrick Kondo, "Monetary Cost-Aware Checkpointing and Migration on Amazon Cloud Spot Instances" , Ieee Transactions On Services Computing
[8] Namjoshi, J.; Gupte, A, "Service Oriented Architecture for Cloud Based Travel Reservation Software as a Service", 2011 IEEE 4th International Conference on cloud computing
[9] Predicts 2007: Software as a Service Provides a Viable Delivery Model. 2006 Gartner, Inc
[10] B. Benatallah, et al., "Facilitating the rapid development and scalable orchestration of composite web services," Distributed and Parallel Databases, vol. 17, no. 1, 2005, pp. 5-37
[11] Lianru Liu; Meina Song; Xiaoxiang Luo; XinHua, E.; Lin Qiu, "A SEDA-based framework for building big web service", 5th International Conference on Pervasive Computing and Applications (ICPCA), 2010
[12] Fu, C.; Belqasmi, F.; Glitho, R." RESTful web services for bridging presence service across technologies and domains: an early feasibility prototype", Communications Magazine, IEEE
[13] Bedra, A.; , "Getting Started with Google App Engine and Clojure," Internet Computing, IEEE , vo1.14, noA, pp.85-88, July-Aug. 2010
[14] Subramanian, V.; Liqiang Wang; En-Jui Lee; Chen, P., "Rapid Processing of Synthetic Seismograms Using Windows Azure Cloud", 2010 IEEE CloudCom
[15] Devi Prasad Bhukya, Prof S. Ramachandram, "Performance evaluation of virtualization and Non virtualization on different workloads using DOE Methodology", IACSIT, International Journal of Engineering and Technology, Vol. 1, No 5, December 2009.
[16] Devi Prasad Bhukya, Prof S Ramachandram, E Chaitra, Reeta Sony A.L, A Novel Methodology for Benchmarking the Hypervisors overHetero geneous Workloads – Multi Vari Approach, , International Journal of Computer Applications (0975 – 8887), Volume 4 – No.4, July 2010
[17] Devi Prasad Bhukya, Prof S Ramachandram, Reeta Sony A.L, "Analysis of Sieve of Eratosthenes method using MPI program over grid and cluster platforms using stastical DOE methodology", IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), 28-29 Dec. 2010