ISSN:0975-9646

Cloud Computing: Designing Different System Architecture Depending On Real-World Examples

Dr. Ravish Saggar[*1], *Ms. Shubhra Saggar*[#2], *Ms. Nidhi Khurana*[#3]

[*1]*Dean, BCIIT, New Delhi*
[#2-3]*Associate professor, GNIM, New Delhi*

Abstract- We decided to focus on the categorization of cloud architectures by looking at the deployment and service models used to describe cloud computing. Researching which architectures are available is hard to do because cloud computing offerings are emerging at a rapid pace. It is not only hard to keep track of all the cloud computing offerings, the cloud computing paradigm itself is also rapidly evolving. Finding out which services are offered and deriving which architectures are used for each of these services, is an impossible goal to achieve in a market this big and developing this fast. As a result, creating a mapping of which security controls are implemented by each cloud provider is a goal aimed too high as well.

1. CLASSIC ARCHITECTURE OF CLOUD COMPUTING

When talking about a cloud computing system, it's helpful to divide it into two sections: the front end and the back end. They connect to each other through a network, usually the Internet. The front end is the side the computer user, or client, sees. The back end is the "cloud" section of the system[15].

The front end includes the client's computer (or computer network) and the application required to access the cloud computing system. Not all cloud computing systems have the same user interface. Services like Web-based e-mail programs leverage existing Web browsers like Internet Explorer or Firefox. Other systems have unique applications that provide network access to clients.

On the back end of the system are the various computers, servers and data storage systems that create the "cloud" of computing services. In theory, a cloud computing system could include practically any computer program you can imagine, from data processing to video games. Usually, each application will have its own dedicated server.

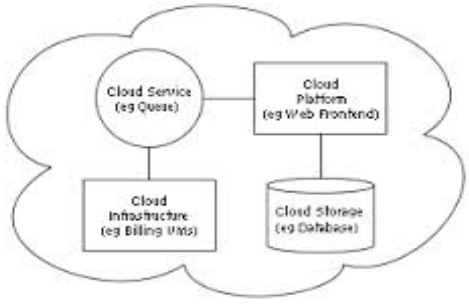

Figure1.Classic Architecture of cloud computing

A central server administers the system, monitoring traffic and client demands to ensure everything runs smoothly. It follows a set of rules called protocols and uses a special kind of software called middleware. Middleware allows networked computers to communicate with each other. Most of the time, servers don't run at full capacity. That means there's unused processing power going to waste. It's possible to fool a physical server into thinking it's actually multiple servers, each running with its own independent operating system. The technique is called server virtualization. By maximizing the output of individual servers, server virtualization reduces the need for more physical machines.

If a cloud computing company has a lot of clients, there's likely to be a high demand for a lot of storage space. Some companies require hundreds of digital storage devices. Cloud computing systems need at least twice the number of storage devices it requires to keep all its clients' information stored. That's because these devices, like all computers, occasionally break down. A cloud computing system must make a copy of all its clients' information and store it on other devices. The copies enable the central server to access backup machines to retrieve data that otherwise would be unreachable. Making copies of data as a backup is called redundancy.

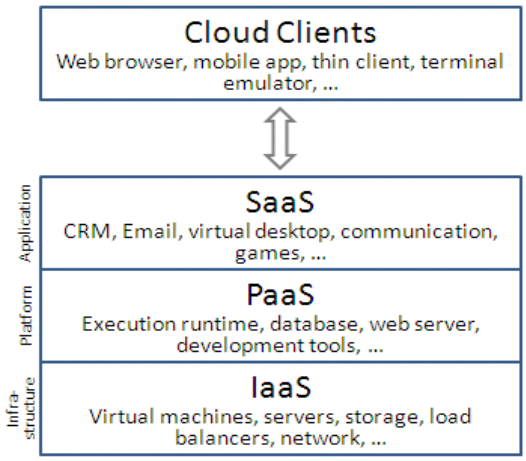

Figure 2 Service models

Cloud architecture, the systems architecture of the software systems involved in the delivery of cloud computing, typically involves multiple cloud components communicating with each other over a loose coupling mechanism such as a messaging queue.

- SaaS applications are designed for end-users, delivered over the web.
- PaaS is the set of tools and services designed to make coding and deploying those applications quick and efficient.

- IaaS is the hardware and software that powers it all – servers, storage, networks, operating systems.

2. ESSENTIAL CHARACTERISTICS OF CLOUD COMPUTING

As described above, there are 5 essential characteristics of Cloud Computing which explains there relation and difference from the traditional computing.

2.1 On-demand-self-service

Consumer can provision or un-provision the services when needed, without the human .

2.2 Broad Network Access

It has capabilities over the network and accessed through standard mechanism.

2.3 Resource Pooling

The computing resources of the provider are pooled to serve multiple consumers which are using a multi-tenant model, with various physical and virtual resources dynamically assigned, depending on consumer demand.

2.4 Rapid Elasticity

Services can be rapidly and elastically provisioned.

2.5 Measured Service

Cloud Computing systems automatically control and optimize resource usage by providing a metering capability to the type of services (e.g. storage, processing, bandwidth, or active user accounts)

3. DATA LOCATION CONCEPT

The exact location of data in the cloud is often unknown. Data may be located in systems in other countries, which may be in conflict with regulations prohibiting data to leave a country or union. It is the responsibility of cloud providers to keep data in specific jurisdictions and whether the providers will make contractual commitments to obey local privacy requirements on behalf of their customers .[6] The location of personal data in relation to the data owner is categorized into three spheres:

3.1 User sphere: location of data is fully controllable by a user, the user is responsible. Data owner Sphere, where the data owner has full control over who accesses his data. In this sphere, the data owner has full control over the information system in which the data is located, and as such can influence the information system infrastructure to support recommended security controls. The Data is stored in Users desktop PCs, laptops, mobile phones, RFID chips.

3.2 Recipient sphere: company-centric sphere of control, control lies with the company. Recipient Sphere, which is an external party-centric sphere of data control in which data owners have no direct control over their data. Any data recipients: servers and databases of network providers, service providers or other parties with whom data recipient shares data

3.3 Joint sphere : companies hosting people‟s data and providing services. Users and providers have a joint control about access to data. Joint Sphere, where a second party hosts the information system and the data, but where the provider and data owner have a joint say as to the degree of access allowed to the data.

4. THINGS TO CONSIDER

There are several factors that you need to take into consideration before designing your own cloud-based systems architecture, particularly if you're considering a multi-cloud/region architecture.[9]

4.1 Cost - Before you architect your site/application and start launching servers, you should clearly understand the SLA and pricing models associated with your cloud infrastructure(s). There are different costs associated with both private and public clouds. For example, in AWS, data transferred between servers inside of the same datacenter (Availability Zone) is free, whereas communication between servers in different datacenters within the same cloud (EC2 Region) is cheaper than communication between servers in different clouds or on-premise datacenters.

4.2 Complexity - Before you construct a highly customized hybrid cloud solution architecture, make sure you properly understand the actual requirements of your application, SLA, etc. Simplified architectures will always be easier to design and manage. A more complex solution should only be used if a simpler version will not suffice. For example, a system architecture that is distributed across multiple clouds (regions) introduces complexity at the architecture level and may require changes at the application level to be more latency-tolerant and/or be able to communicate with a database that's migrated to a different cloud for failover purposes.

4.3 Speed - The cloud gives you more flexibility to control the speed or latency of your site/application. For example, you could launch different instance types based on your application's needs. For example, do you need an instance type that has high memory or high CPU? From a geographic point of view which cloud will provide the lowest latency for your users? Is it necessary or cost effective to use a content distribution network (CDN) or caching service? For user-intensive applications, the extra latency that results from cross-cloud/region communication may not be acceptable.

4.4 Cloud Portability - Although it might be easier to use one of the cloud provider's tools or services, such as a load balancing or database service, it's important to realize that if and when you need to move that particular tier of your architecture to another cloud provider, you will need to modify your architecture accordingly. Since ServerTemplates are cloud-agnostic, you can use them to build portable cloud architectures.

4.5 Security - For MultiCloud system architectures, it's important to realize that cross-cloud/region communication is performed over the public Internet and may introduce security concerns that will need to be addressed using some type of data encryption or VPN technology.

5. PROGRESSION FROM SIMPLE TO MORE COMPLEX REFERENCE ARCHITECTURES:

The architecture diagrams below show a progression from simple to more complex reference architectures.

5.1 Single "All-in-one" Server

Use one of the "All-in-one" ServerTemplates, such as the LAMP (Linux, Apache, MySQL, PHP) ServerTemplate

to launch a single server that contains a web server (Apache), as well as your application (PHP) and database (MySQL).You'll find a collection of simple "All-in-one" ServerTemplates in the MultiCloud Marketplace, which are useful for new users and basic demos.

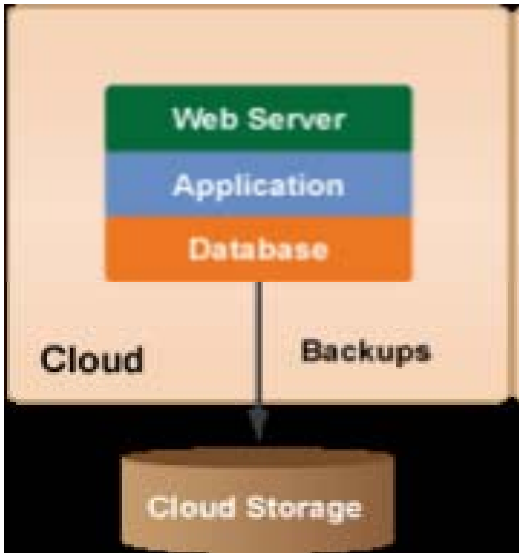

Figure 3 Single "All-in-one" Server

5.2 Single Cloud Site Architectures

In a standard three-tier website architecture, there is at least one dedicated server in each tier of the system architecture. (Load Balancing Server, Application Server, Database Server)

5.2.1 Non-Redundant 3-Tier Architecture

If you are only testing the interactivity between each tier of your architecture, you may want to use a non-redundant system architecture to save on costs and resources. Since it is a non-redundant system architecture it is primarily used for basic test and development purposes. In the example diagram below, there are dedicated servers for each tier of the application/site. A non-redundant architecture is not recommended for production environments.

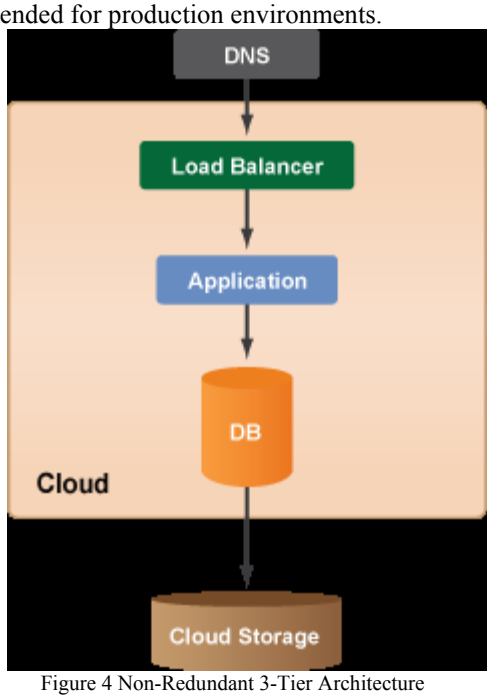

Figure 4 Non-Redundant 3-Tier Architecture

5.2.2 Redundant 3-Tier Architecture

Any production environment that is launched in the cloud should also have a redundant architecture for failover and recovery purposes. Typically, you will use a Server Array for your application tier to take advantage of auto scaling in the cloud, however there may be some scenarios where your application is not designed to auto scale. In such cases, you

can still create a redundant multi-tier architecture where you have redundancy at each tier of your reference architecture. In the example below, there are two load balancer servers, two application servers, as well as master and slave database servers. A redundant architecture will help protect your site/application from system downtime.

This example diagram also demonstrates the use of a striped volume set at the database tier. If your database is large and requires faster backups, you may consider using a set of striped volumes for data storage.

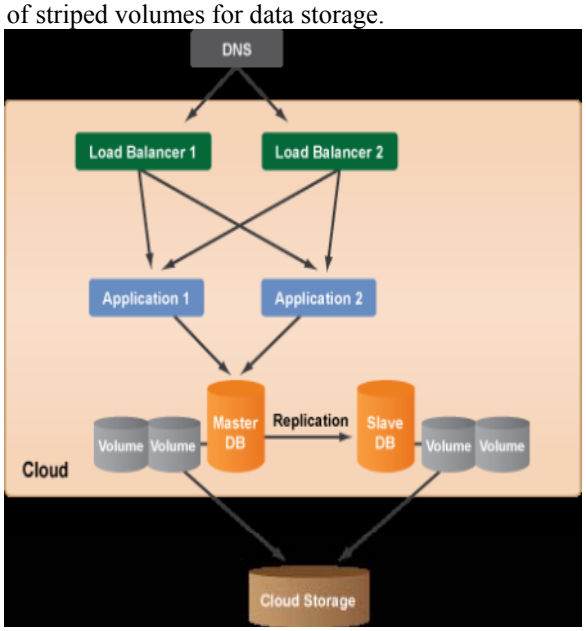

Figure 5 Redundant 3-Tier Architecture

5.2.3 Multi-Datacenter Architecture

If your cloud infrastructure supports multiple datacenters (or zones), it's recommended that you spread your system architecture across multiple datacenters to add another layer of redundancy and protection. Each datacenter in a cloud is designed to be an isolated segment inside the same geographical cloud. So if a power failure occurs in one datacenter, the other datacenters will be unaffected. For example, within a cloud/region there may be several resource pools called availability zones and datacenters. The benefit of using multiple datacenters is to protect your entire site/application from being negatively affected by some type of network/power failure, lack of available resources, or service outtage that's specific to a particular datacenter.

As a best practice you should always leverage multiple datacenters in your reference architecture if they are supported by the cloud infrastructure. In the other reference architecture diagrams below, it's also recommended that you usemultiple datacenters even though it's not explicitly shown in the diagrams.

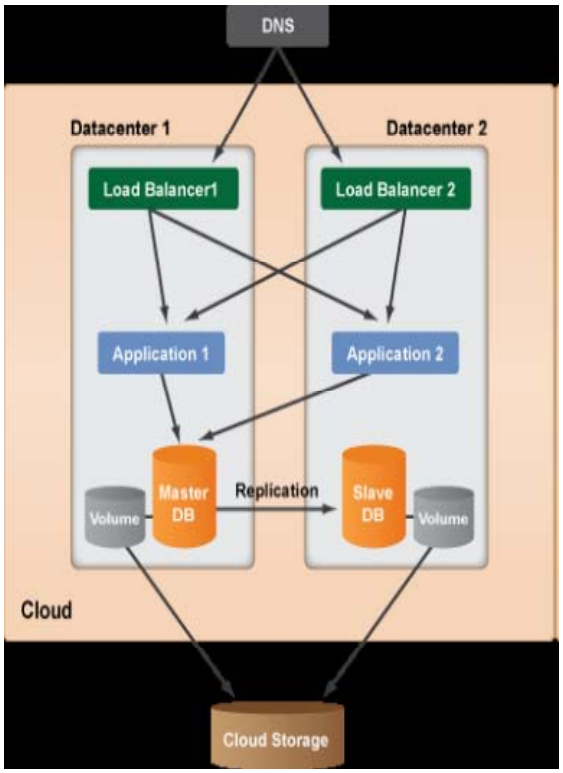

Figure 6 Multi-Datacenter Architecture

5.2.4 Autoscaling Architecture

One of the key benefits of the cloud is the ability to horizontally scale (i.e. grow or shrink the number of running server resources) as the demands of your application/site change over time. You can use Server Arrays to set up a particular tier of your architecture to autoscale based on predefined alert conditions. Autoscaling is most commonly used for the application tier of your cloud reference architecture.

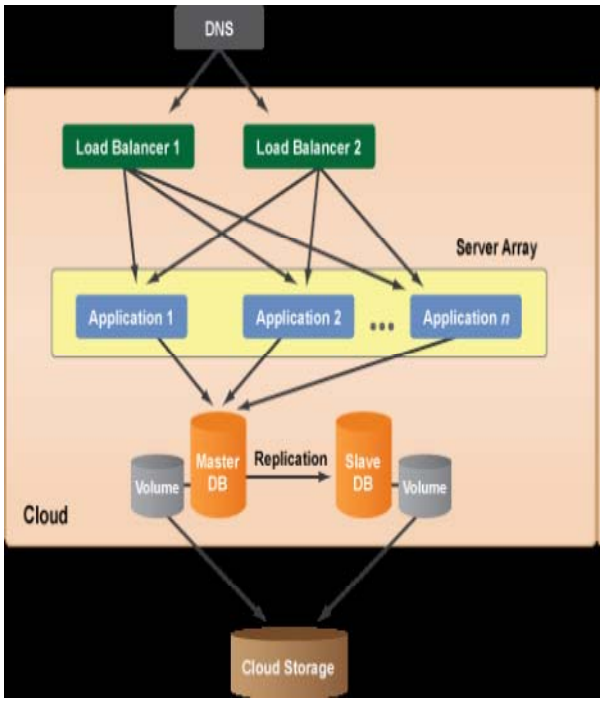

Figure 7 Autoscaling Architecture

5.3 Hybrid Cloud Site Architectures

Another way that you can protect your site/application in the cloud is to design a hybrid cloud site architecture that leverages multiple public/private cloud infrastructures or dedicated hosted servers. One of the key benefits of this platform is cloud portability, where you can use the same assets (ServerTemplates, RightScripts, etc.) to launch identically functioning servers into multiple public/private clouds. Avoid cloud lock-in and design a solution architecture that takes advantage of multiple cloud resource pools instead of just a single cloud. Similarly, you can also design a hybrid cloud architecture where servers in a cloud can communicate with dedicated servers that are hosted in an internal/external datacenter.

5.3.1 Scalable MultiCloud Architecture

In the example below, we are using one cloud infrastructure to host your site/application, but you've also set up a Server Array for autoscaling your application tier in a different cloud infrastructure. For example, you might use your own private cloud servers before incurring any costs associated with launching servers in public cloud infrastructures. The MultiCloud Architecture diagram below gives you the flexibility of primarily hosting your application in your private cloud infrastructure but also autoscale out into a public cloud for additional server capacity, if necessary.

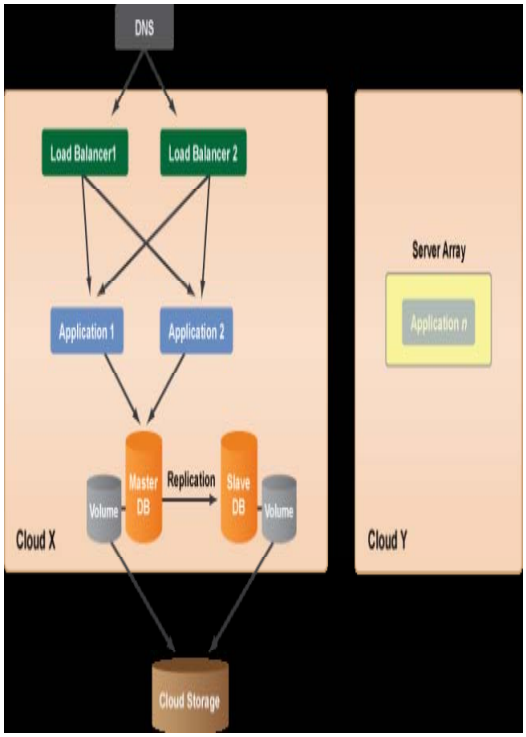

Figure 8 Scalable MultiCloud Architecture

5.3.2 Failover MultiCloud Architecture

In the example diagram below, the same ServerTemplates and scripts are used to configure and launch functional servers into either Cloud X or Y. When you are designing your cloud system architecture across multiple clouds, there are several factors that you will have to take into consideration. In the example below, there is a running Slave-DB server that's serving as a "warm"

backup, but it's replicating data with the Master-DB across the public, not private IP address. Remember, only servers within the same cloud infrastructure can communicate over a private IP address.

However, if there is ever a problem or failure that would require you to switch clouds, a MultiCloud Architecture would allow you to easily migrate your site/application. Notice that the other tiers of the reference architecture have already been configured and are ready to be launched if you need to migrate your production environment from Cloud X to Cloud Y. It's important to remember that the clouds could be any combination of public/private cloud.

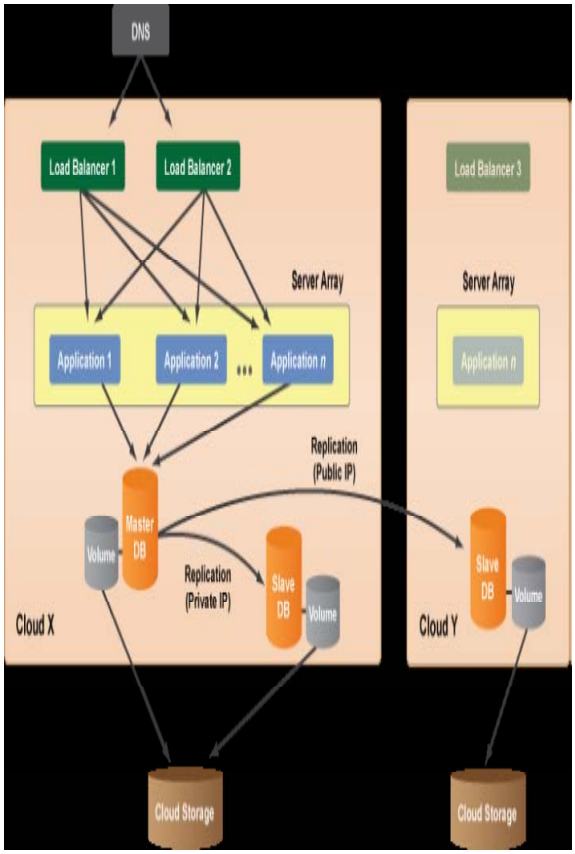

Figure 9 Failover MultiCloud Architecture

6. CONCLUSION

The purpose of the system architecture diagrams above is to provide you with real-world examples that you can use as base reference architectures when you design your own custom system architectures in the cloud. Once you find a system architecture that is similar to what you are trying to build, you can modify and customize it accordingly to meet your own project's requirements. The diagrams are designed to demonstrate a particular concept such as disaster recovery or multicloud deployments. When you are designing your own solution architectures you should consider integrating several of the concepts described above.

Cloud environments can also be categorized with the deployment model, which describes first of all which party owns the infrastructure, secondly which party manages the infrastructure, and finally at whose location the infrastructure is located.

REFERENCE

[1] Gens, F.: IT Cloud Services User Survey, part 2: Top Benefits and Challenges(2008)

[2] John D. Sutter, Twitter Hack Raises Questions About "Cloud Computing," CNN, July 16, 2009, http://www.cnn.com/2009/TECH/07/16/twitter.hack/index.html.

[3] Amazon. Amazon Elastic Compute Cloud(EC2),2010 /http://www.amazon.com/ ec2/S [accessed: 10December2009].

[4] Bernard Golden.Definingprivateclouds,2009 /http://www.cio.com/article/492695/Defining_Private_Clouds_Part _OneS [accessed on:11January2010].

[5] Boss ,MalladiP,QuanD,LegregniL,HallH.Cloudcomputing,2009,p.4 http://www.ibm.com/developerswork/websphere/zones/hipods/library.htmlS [accessedon:18October2009].

[6] Michael Miller, "Cloud Computing: Web-Based Applications That Change the Way You Work and Collaborate Online", August 2008

[7] Peter Mell, and Tim Grance, "The NIST Definition of Cloud Computing," 2009, http://www.wheresmyserver.co.nz/storage/ media/faq-files/clouddef-v15.pdf, Accessed April 2010.

[8] Frank Gens, Robert P Mahowald and Richard L Villars. (2009), IDC Cloud Computing 2010.

[9] http://sushilresearch.wordpress.com/2013/05/04/cloud-computing

[10]. Jason Bloomberg, "Data Remanence: Cloud Computing Shell Game," May 19, 2011. http://www.zapthink.com/2011/05/19/data-remanence-cloud-computing-shell-game/.

[11]. Amazon EC2 goes down, taking with it Reditt, FourSquare and Quora. http://eu.techcrunch.com/2011/04/21/amazon-ec2-goes-down-taking-with-it-reddit-foursquare-and-quora/.

[12] Rajkumar Buyya, Chee Shin Yeo, and Srikumar Venugopal, "Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities," Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications (HPCC 2008), Dalian, China, Sept. 25-27, 2008.

[13] Zhenyu Fang and Changqing Yin, "BPM Architecture Design Based on Cloud Computing," Online Journal on Intelligent Information Management, Vol 2, May 2010, pp 329-333. [3] Norman W. Paton, Marcelo de Aragao, Kevin Lee, Alvaro Fernandes and Rizos Sakellariou, "Optimizing Utility in Cloud Computing through Autonomic Workload Execution," retrieved from http://research.microsoft.com/pub/debull/A09mar.

[14] Sean K Barker, Prashant Shenoy, "Empirical Evaluation of Latency-sensitive Application Performance in the Cloud.

[15] http://computer.howstuffworks.com/cloud-computing/cloud-computing1.htm

[16] / IDC, "IDC Ranking of issues of Cloud Computing model," ed, 2009,http://blogs.idc.com/ie/?p=210, Accessed on July 2010.