Design of QoS Architecture for Global Cloud Computing Services by Adaptive Scheduling Algorithms

K. Ramkumar
Research Scholar
Manonmaniam Sundaranar University
Computer Science and Engineering
Indira Institute of Engineering & Technology
Pandur, Thiruvallur - 631203, Chennai, India.

G.Gunasekaran, Ph. D
Anna University
Computer Science and Engineering
Meenakshi College of Engineering
West KK Nagar, Chennai-600078, India.

ABSTRACT
In this paper we will design architecture for cloud computing services with QoS by supported scheduling algorithm for resource allocation. Cloud generally uses virtualization technology which is specially designed for maximum resource utilization. The QoS of cloud environment for any kind of resource utilization is achieved feasibly by using QoS-aware Cache Replacement algorithm. This algorithm makes maximum utilization of resource in Cloud Environment. This accuracy is investigated by us using some measurements such as simple ping measurements which can be used as an indicator for some other QoS parameters such as jitter, throuput and delay. As a result, we were able to monitor the changes in QoS parameters during a number of days. This is also a first step towards defining QoS parameters to be included in Service Level Agreements for cloud computing in the foreseeable future.

General Terms
QoS-aware Cache Replacement algorithm, Service Level Agreements, virtualization

Keywords
Adaptive scheduling, Cache Replacement Algorithm, QoS, Resource Management

1. INTRODUCTION
In this paper, we deal about how to design an adaptive QoS management framework for the VoD cloud service centers. The main contributions of the paper include: I) the designation of QoS management framework mainly followed following ways such that on the concept of autonomic computing 2) the development of the QoS-aware Cache Replacement algorithm which is aiming at maximizing SLA-based profits 3) experiments based on a prototype system and simulation, demonstrating the feasibility and efficiency of proposed approaches. Finally the QoS parameter is verified by means simple ping measurement.

Although the cloud-based model is well suitable into the requirements of media streaming over the Internet, cloud media delivery is always a challenging area because of its bandwidth requirement and timing constraint. Media applications especially video conferencing have the requirement of low end-to-end latency. With the network traffic also increases, the quality of service (QoS) of multimedia applications degrades and finally completely falling down. It suffers from reduced video quality or increased delay and jitter. Currently there is not much research done on the QoS provision of media applications in the cloud computing environment. A general description of QoS of applications in cloud computing is addressed in [4]. Multimedia-aware cloud platforms are proposed in [6] and [5] to provide QoS for multimedia processing and storage. An IP multimedia subsystem with cloud computing architecture was proposed in [7] to allow the users to access multimedia application via Android-based smart devices. The proposed design provides video streaming services with server virtualization technology.

In order to smoothly connect to services to clients offered from servers with other clients located all over the globe, a specified Quality of Service (QoS) is often required, this is sometimes expressed as in Service Level Agreements (SLAs). If these servers and clients are just connected through the Internet, it can be challenging to obtain a consistent service: The traffic is dynamically routed through different providers, from end-user access at the edge through distributed networks to national or international. This also makes it difficult to provide guarantees or even predictions of QoS since most often we do not have insight into the provider's networks and routes of global connections which are likely to change dynamically and continuously. Known behaviors, such as temporal changes in traffics, may also be different networks to networks, making it difficult to make up with a prediction model. Moreover, different kinds of traffic may be prioritized based on e.g. packet sizes, protocols and source/destination addresses, adding to the complexity of modeling and predicting behaviors, or even continuously monitoring changes in QoS in a simple manner. Working in this uncontrolled environment also makes it hard to apply existed QoS techniques which focus on providing guarantees based on admission control ability[8][9].

Cloud Computing (CC) technology and its supporting services are now a day's regarded as a hottest trend move towards distributed and pervasive computing services which is offered over the global resources via Internet. Several architectures exist for CC, which behaves differ in what kind of computing services are offered to this environment, culminating with the advances with Web 3.0 and Web 4.0 technologies [1]. Detailed studies of different approaches to CC can be found in [2][3]. To keep it simple, CC can be divided into two domains. The first domain consists of resources for computations and

applications access, and their use by users – seen as traditional client server model.

2. BACKGROUND AND DEFINITIONS

The main QoS parameters are throughput, delay, jitter and packet loss. Since the work in this paper mainly deals with high speed-capacity connections between clouds, the focus is on the following three parameters, all relevant to different real-time and non-real-time CC services:

Throughput*:* Measured as the average maximum data rate from source to destination when transmitting a file. Thus, in our terms throughput is measured unidirectional at a time. For the experiments in this paper, it is measured by the time it takes to transmit a file of a certain length.

Delay*:* Measured as the round trip time for packets, simply using the standard Ping command.

Jitter*:* Measured based on variation only in delay. The measurements on the Ping packets as explained above, and adopts the definition taken from RFC 1889 [12]: $J=J'+(|D(i-1,i)|-J')/16$. So the jitter J is calculated continuously every time a Ping packet is received, based on the previous jitter value, J', and the value of $|D(i,j)|$ which is the difference in Ping times between the *i'th* and *j'th* packets.

Packet loss is not considered during this study, since it is expected to be statistically not necessary (the assumption was actually confirmed during the experiments).

3. CLOUD COMPUTING ARCHITECTURE

Cloud computing is an umbrella term used to refer to Internet based development and services. The cloud is a metaphor for the Internet. A number of characteristics define cloud data, applications services and infrastructure.

Remotely hosted: Services or data are hosted on someone else's infrastructure.

Ubiquitous: Services or data are available from anywhere.

Commodified: The result is a utility computing model similar to traditional that of traditional utilities, like gas and electricity. You pay for what you would like.

3.1 Software as a Service (SaaS)

SaaS is a model of software deployment where an application is hosted as a service provided to customers across the Internet. SaaS is generally used to refer to business software rather than consumer software, which falls under Web 2.0. By removing the need to install and run an application on a user's own computer it is seen as a way for businesses to get the same benefits as commercial software with smaller cost outlay. (SaaS) also alleviates the burden of software maintenance and support but users relinquish control over software versions and requirements. The other terms that are used in this sphere include Platform as a Service (PaaS) and Infrastructure as a Service (IaaS).

3.2 Cloud Storage

Several large Web companies (such as Amazon and Google) are now exploiting the fact that they have data storage capacity which can be hired out to others. This approach, known as 'cloud storage' allows data stored remotely to be temporarily cached on desktop computers, mobile phones or other Internet-linked devices. Amazon's Elastic Compute Cloud (EC2) and Simple Storage Solution (S3) are well known examples.

3.3 Data Cloud

Cloud Services can also be used to hold structured data. There has been some discussion of this being a potentially useful notion possibly aligned with the Semantic Web [2], though concerns, such as this resulting in data becoming undifferentiated [3], have been raised.

3.4 Opportunities and Challenges

The use of the cloud provides a number of opportunities:

• It enables services to be used without any understanding of their infrastructure.

• Cloud computing works using economies of scale. It lowers the outlay expense for startup companies, as they would no longer need to buy their own software or servers. Cost would be by on-demand pricing. Vendors and Service providers claim costs by establishing an ongoing revenue stream.

• Data and services are stored remotely but accessible from 'anywhere'.

In parallel there has been backlash against cloud computing:

• Use of cloud computing means dependence on others and that could possibly limit flexibility and innovation. The 'others' are likely become the bigger Internet companies like Google and IBM who may monopolies the market. Some argue that this use of supercomputers is a return to the time of mainframe computing that the PC was a reaction against.

•Security could prove to be a big issue. It is still unclear how safe outsourced data is and when using these services ownership of data is not always clear.

• There are also issues relating to policy and access. If your data is stored abroad whose FOI policy do you adhere to? What happens if the remote server goes down? How will you then access files? There have been cases of users being locked out of accounts and losing access to data.

4. PROPOSED CLOUD COMPUTING ARCHITECTURE

The proposed cloud computing architecture includes storage, software, networking, virtualization and resource management and efficient QOS ability. Since cloud clients can also access the services via heterogeneous network connections with various devices, bandwidth

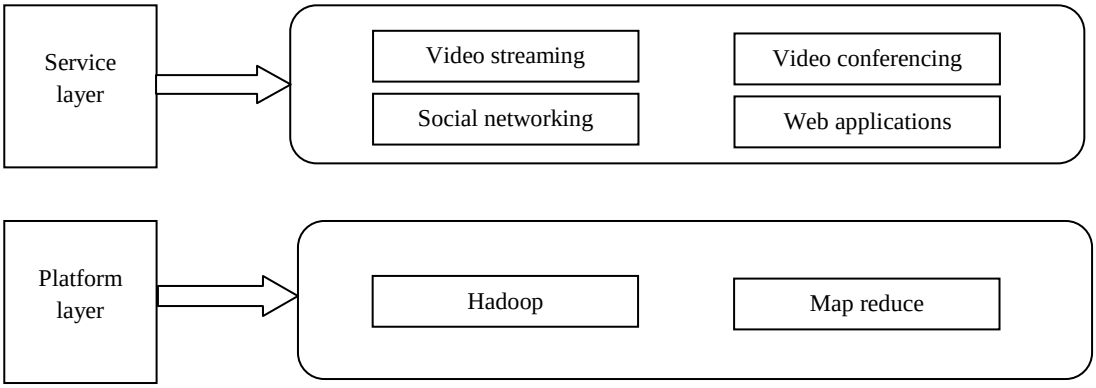

Figure 1 Proposed Cloud Architecture

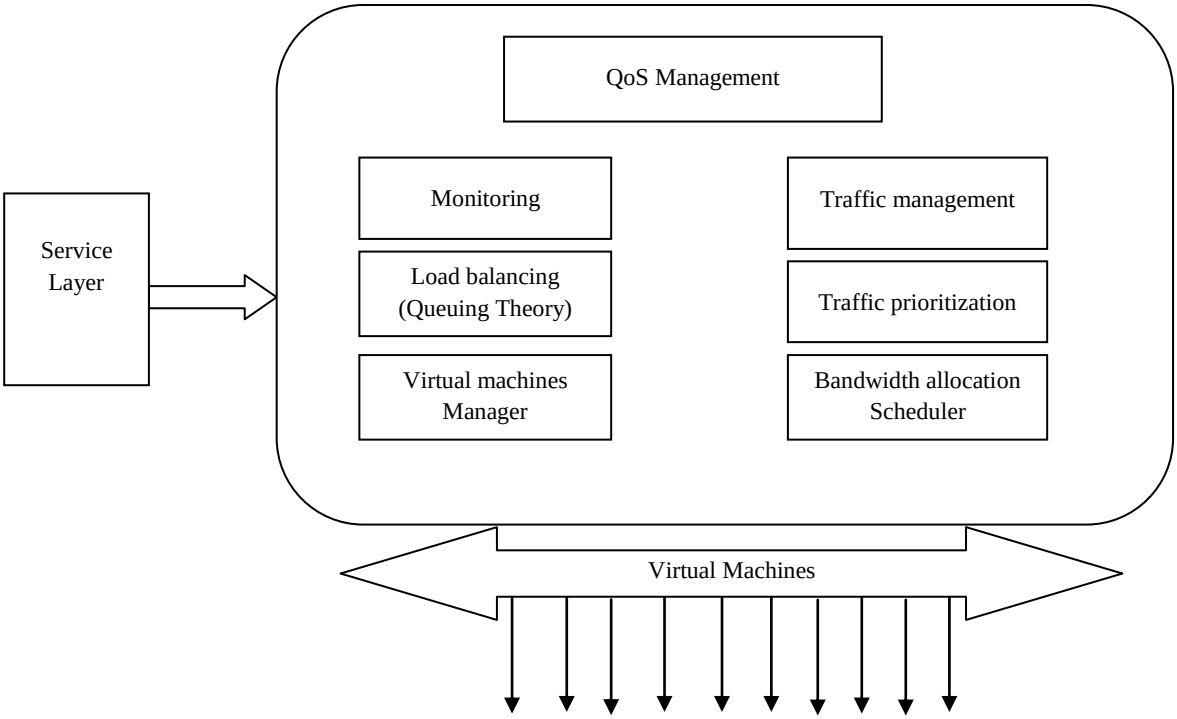

Figure 2 Basic Cloud Architecture

Allocation is crucial to the QoS of sending data to provide the end-to-end delay guarantee. This architecture consists of three basic layers: infrastructure layer, platform layer and service layer. Fig. 1 displays the proposed architecture and its components.

At the bottom of the architecture, multiple virtual machines can be created at the same physical machine to share the resources of the hardware. There can be a possibility of multiple platforms running in a single physical machine. The design for QoS based cloud computing is implemented at the infrastructure layer. Three classes of service are defined namely guaranteed service, differentiated service and best effort service. Applications can subscribe to any one of the three classes of service according to their requirements gathered as request. For guaranteed service, the infrastructure reserves the resources for applications to guarantee the delay. For differentiated service provision, the framework is able to allocate resources to the media traffic according to the service level and the priorities of the queue. Best effort service is provided to application with low priority always.

The major components for the provision of QoS are described as below,

• **Monitoring**: The framework monitors the resource accessing ratio and the performance, such as the bandwidth consumption, latency, server load, memory usage, and CPU utilization. The collected information will be used as feedback to load balancing and traffic management using priority.

• **Load balancing**: Load balancing determines the best physical server and virtual server to host the application so all the servers can achieve the same level of utilization. Via load balancing, applications can be ensured service

with given levels of performance. Load balanced is achieved by means of queuing theory.

• **Traffic management**: Traffic is classified as under traffic prioritization according to the corresponding service class and SLAs. Bandwidth allocation and scheduling are used to allot the networking resources to the traffic. The allotment of resources should be good to the bandwidth of the access networks.

• Security: It provides the security for the infrastructure, platform and software, which includes the identity and access management.

At the platform layer, the combination of Hadoop [14] and Map Reduce is adopted to provide the platform for PaaS service. Hadoop provides the software framework to support data-intensive distributed applications with high scalability with efficiency. At the service layer, various applications, such as video conferencing and social networking, are running to provide good services for end users.

5. QoS-AWARE RESOURCE MANAGEMENT STRATEGIES

Given an established VoD cloud service center, the hardware configuration will be fixed, including the number of disks, the size of memory of disks and also the storage size of the videos. Thus, in order to maximize the total profits, here we focus on the consideration of resource management strategies, namely that how to allocate limited resource to different classes of users.

Since a brute-force method is computationally expensive, here we will employ heuristic algorithms to find good solution for the problem. A basic idea is to prioritize the QoS guarantee for higher-class users, which is charged more for better service quality. For instance, among the incoming requests, the scheduling algorithm should change the processing order according to their priorities.

Users who demand for a certain video clip usually want to start watching it as soon as possible, with as less latency as possible. However, loading a file from disks is often much slower than from the memory cache. Ideally, if all the videos are located in memory, the service efficiency will be greatly enhanced. Nevertheless, the memory size is limited compared to the disk storage size, and thus appropriate cache replacement algorithms are needed to guarantee the QoS delivered to users.

Assume that the user-concern QoS parameter is the accumulated latency time from the start to the end of watching. Hence, the service provider should try to lower down the latency to increase the profit. In order to decide whether a video block bi should be cached or not, a cache gain value is calculated as the sum of access frequency gain and predicted revenue gain, namely

$$g(b_i) = w_f * g_f(b_i) + w_r * g_r(b_i)$$

$$g_f(b_i) = f(b_i)$$

$$gr(bi) = \sum_{j=0}^{nu} (\max((tbi - tuj), 0) * ruj)$$

where f(bi) is the access frequency of block bi in the last time period; nu is the number of current waiting users; tbi is the start offset time of block bi relative to the original video file; tuj. is the specified start time of request Uj; and ruj. is the predicted revenue brought by finishing requests Uj, based on the current running status of the service.

Accordingly, we propose the QoS-aware Cache Replacement (QCR) algorithm, and the formal description is illustrated as Fig. 3. As illustrated, there is a list recording the gain values of each block cached in the memory. A gain value is updated once the block has been accessed. If the memory size is insufficient to cache new blocks, the old block with the lowest gain value will be replaced. According to the QoS-aware strategy, blocks needed by higher-class users tends to have larger gain values than those needed by lower-class users.

Thus, the total revenue could be increased and the QoS could be guaranteed from time to time while the blocks located in the memory are dynamically adjusted under the control of cache replacement strategies.

QoS-aware Cache Replacement Algorithm:

Each time a video block hi is accessed, the algorithm will update the new gain value of this block using the following method

```
UpdateBlockGain(DataBlock bi))
{
f=GetAccessFrequency(bi);
f=f+1;
gain=O;
for each req_i in the RequestList
{
if(b_i is part of the video file req_j orders)
{
offset= b_i.offsetTime- req_j.startTime;
if( offset < 0)
offset=0;
gain+=offset*predictedRevenue(req_i);
gain=w_r*gain+w_t*f;
UpdateAccessFrequency(b_i, f);
if (b_i is in blockGainList)
update(blockGainList, b_i, gain);
else
insert(blockGainList, b_i, gain);
}
```

Test Setup

First the possible correlation between latency and throughput is investigated, based on the results shown in Figures 4-6. In order to be able to observe temporal behaviors, the measurement values are arranged from midnight to midnight, even though the actual experiments are starting at different times as listed in the figure captions. It is hard to find a consistent correlation between the two parameters, and in many places the parameters seem to change independently of each other. This is for example the case for the "spike" of increase in latency in Figure 4, shown in more details in Figure 5. The latency

increases significantly for a while, but this is not matched by an increase in file transfer times. In other of the figures there appear to be a relationship, where an increase in latency also results in increased file transfer times. This is most visible where the latency is quite stable over time; small spikes in latency are matched by spikes in throughput. This is clearly visible in Figures 4, 5 and 6, where the latter shows a more detailed view of the first part of Figure 4, and can also be seen in other figures. The tendency was confirmed also by studying more of the experiments closer. The opposite does not seem to hold: file transfer time seems to vary even when the latency remains constant, and when latency increases this is not necessarily reflected in the file transfer times. What can also be observed from these figures is that there are no consistent variations over the 24 hour periods. The variations are generally locally varying over time, with some rather dramatic changes, for which we do not know the reasons. For the Polish results (Figures 5-6) there could be a relation, where file transfer times and to some extent latency increase during working hours, but it is hard to tell, and the patterns are not really similar for the two days. Calculating correlation coefficients for the relationship between Ping and latency did not lead to conclusive results.

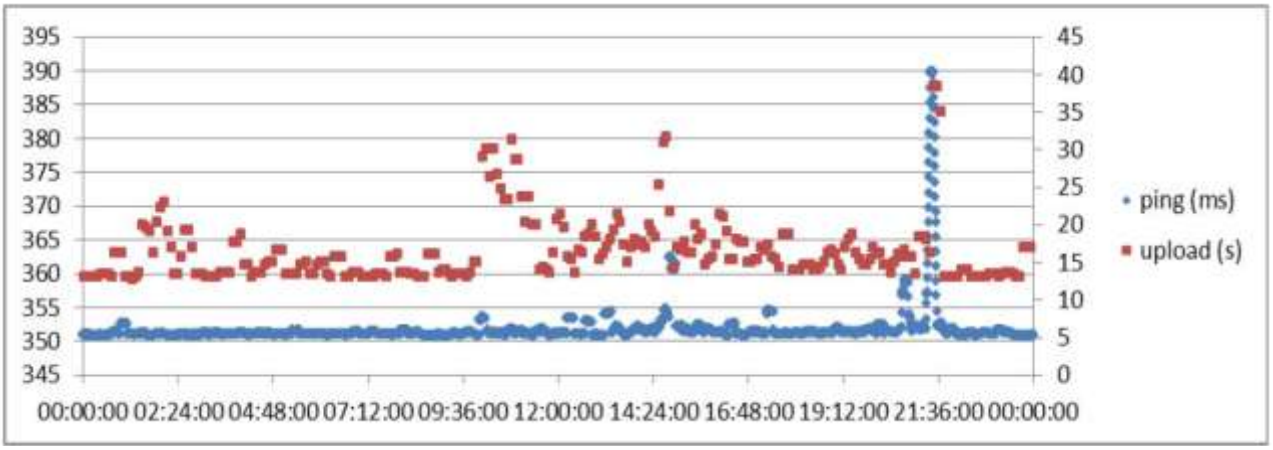

Figure 3: Latency (Ping, left scale) and throughput (upload times, right scale) for the second experiment between AAU and the server in Poland. The experiments were carried out between 10:00 and 10:00 (Danish time, UTC+1).

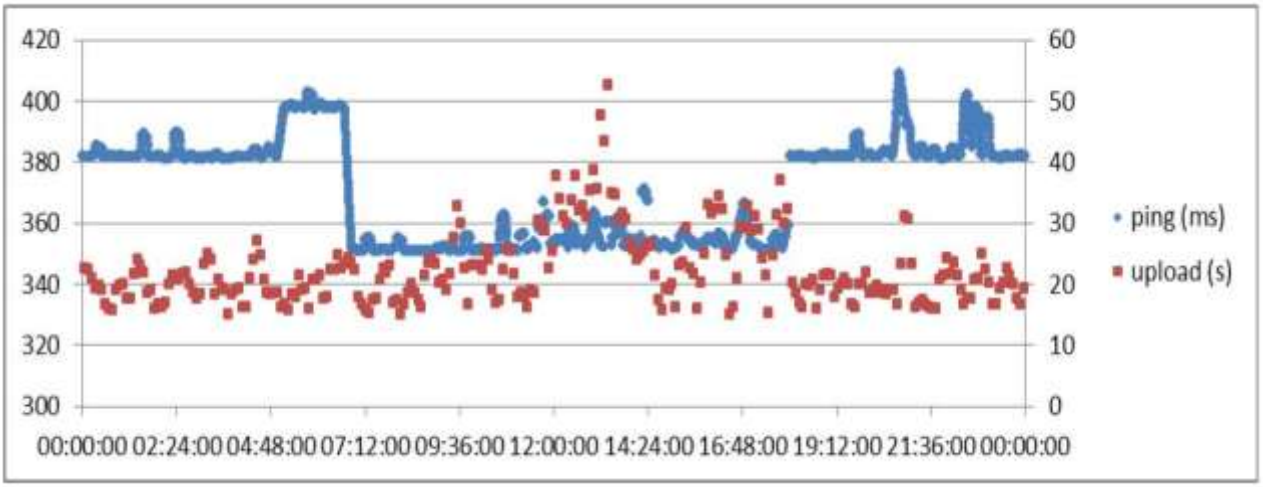

Figure 4: Latency (Ping, left scale) and throughput (upload times, right scale) for the experiment between AAU and the server in Malaysia. The experiments were carried out between 11:30 and 11:30 (Danish time, UTC+1).

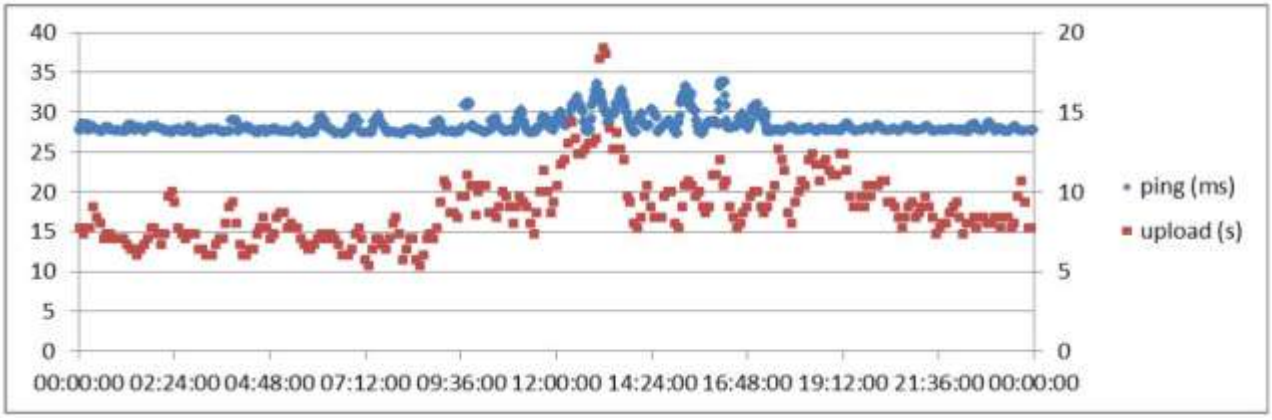

Figure 5: Latency (Ping, left scale) and throughput (upload times, right scale) for the experiment between AAU and the server in Malaysia. The experiments were carried out between 11:30 and 11:30 (Danish time, UTC+1).

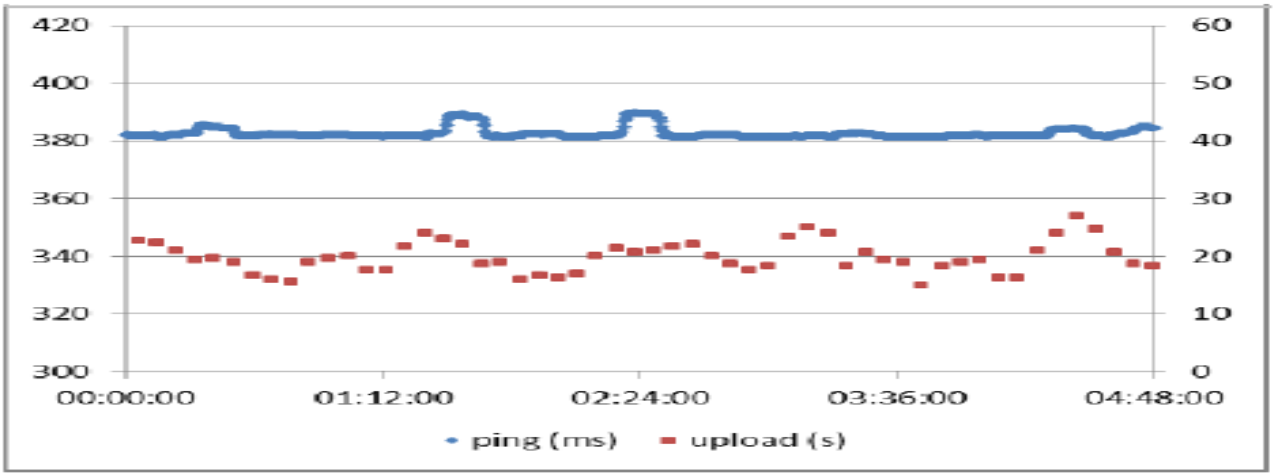

Figure 6: Based on the above results, for the first 4:48 hours, the upload times seem to increase during the latency spikes.

6. TABLES

Table 1

Seconds	Ping	Upload
0 – 21	350	5
2.24 - 4.48	360	10
7.24 – 9.21	365	15
10.12 – 12.45	370	16
14.12-16.12	375	20
17.12-18.12	380	31
19-20	380	35
20 – 21	380	40

Table 2

Seconds	Ping	Upload
0 – 21	350	5
2.24 - 3.48	355	10
5.24 – 9.21	365	15
10.32 – 12.45	370	16
12.62-16.12	375	22
18.12-18.12	385	34
19-20	390	45
20 – 21	400	55

Table 3

Seconds	Ping	Upload
0 – 2.32	5	5
2.44 - 3.48	7	8
4.24 – 5.21	8	9
6.32 – 7.45	9	9
7.62-8.12	15	11
8.12-10.12	16	12
19-20	17	13
20 - 21	26	18

Table 4

Seconds	Ping	Upload
0 – 1	340	20
1.24 - 1.48	345	15
2.24 – 2.27	355	16
2.32 – 3.45	340	15
3.62-3.72	330	16
3.82-3.92	336	16
3.95-4.10	340	17
4.20 – 4.48	355	16

The strongest correlation was found in the first experiment between AAU and Poland, where the correlation coefficient was 0.49. For the other experiments the values were 0.22 (Poland), -0.35 and 0.29 (Brazil) and 0.37 (Malaysia). Next the relationship between latency and jitter is studied. At a first glance, there is a close dependency between latency and jitter, where the spikes in latency is followed also by spikes in jitter. See Figures 4-6. Some relationship was also confirmed by the correlation coefficients, which were respectively 0.69 and 0.33 (Poland), 0.00 and 0.47 (Brazil) and 0.58 (Malaysia).

7. CONCLUSION

This Design is basically well answered in media data transferred. It will have a bit of difference when we send text data. So this design will work well for big data than normal data. Now a day's world is moving towards big data server as the request coming from client and end user is very high now a days. But in current cloud server it is not easily handled in a very superior manner. In future this design will give a very efficient live data to cloud environment.

8. ACKNOWLEDGMENTS

I sincerely thank my Guide and my Professor Dr.G.Gunasekaran for his constant guidance during the preparation of this manuscript. I also thank my fellow professors with their valuable suggestions and ideas from time to time.

9. REFERENCES

[1] L. Na, A. Patel, R. Latih, C. Wills, Z. Shukur, R. Mulla, "A study of mashup as a software application development technique with examples from an end-user programming perspective". Journal of Computer Science 6 (11), November 2010.

[2] I.Foster, Y. Zhao, I Raicu, S. Lu, "Cloud Computing and Grid Computing 360-Degree Compared". In: Grid Computing Environments Workshop, 2008. GCE'08, pp. 1–10.

[3] Qi Zhang, Lu Cheng, Raouf Boutab, "Cloud computing: state-of-theart and research challenges". Journal of Internet Services and Applications, Volume 1, Number 1, pp. 7-18

[4] Amazon EC2, aws.amazon.comlec21

[5] D. Kovachev, R. Klamma, "A Cloud Multimedia Platform", in Proceedings of the 11th International Workshop of the Multimedia Metadata Community on Interoperable Social Multimedia Applications (WISMA-201O), Barcelona, Spain, pp. 61-64. May 19-20, 2010,

[6] W. Zhu, C. Luo, J. Wang, S. Li, "Multimedia Cloud Computing: Directions and Applications", Special Issue on Distributed Image Processing and Communications, IEEE Signal Processing Magazine, Vol. 28, No. 3, pp. 59-69, May 2011.

[7] J.-L. Chen, S.-L. Wu, Y.T. Larosa, P.-J. Yang, Y.-F. Li, "IMS cloud computing architecture for high-quality multimedia applications", in Proceedings of the 7th International Wireless Communications and Mobile Computing Conference (IWCMC), pp. 1463 - 1468, July, 2011

[8] O. T. Brewer, A. Ayyagari, "Comparison and Analysis of Measurement and Parameter Based Admission Control Methods for Quality of Service (QoS) provisioning". Proceedings of the 2010 Military Communications Conference, IEEE 2010.

[9] S. Y. Ban, J. K. Choi, H.-S. Kim, "Efficient End-to-End QoS Mechanism Using Egress Node Resource Prediction in NGN Network". In Proceedings of ICACT 2006, IEEE 2006.

[10] Welcome to the Data Cloud, The Semantic Web blog, 6 Oct 2008, http://blogs.zdnet.com/semantic-web/?p=205

[11] Any any any old data, Paul Walk's blog, 7 Oct 2008, http://blog.paulwalk.net/2008/10/07/any-any-any-old-data/

[12] M. Armbrust, A. Fox, R. Grifth, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. P. A. Rabkin, L Stoica, and M. Zaharia, "Above the Clouds:A Berkeley View of Cloud Computing," Technical report, 2009.

[13] D. Armstrong and K. Djemame. 'Towards Quality of Service in the Cloud", in Proceedings of the 25th UK Performance Engineering Workshop, Leeds, UK, pp. 226-240, July 2009.

[14] Hadoop, http://hadoop.apache.org/