QOS BASED SCHEDULING OF WORKFLOWS IN CLOUD COMPUTING UPNP ARCHITECTURE

[1]K. Ramkumar, [2]Dr.G.Gunasekaran

1Research Scholar, Computer Science and Engineering
Manonmaniam Sundaranar University
Tirunelveli - 627012, Tamilnadu, India.
ramkumar1975@gmail.com
2Anna University, Computer Science and Engineering
Meenakshi College of Engineering
West KK Nagar, Chennai-600078, India.
gunaguru@yahoo.com

ABSTRACT

Cloud computing is a collection of virtualized computers that are probed on-demand to service applications. [1] *These applications are represented as workflows which are a set of tasks processed in a specific order based on their required services. Scheduling these workflows to get better success rate becomes a challenging issue in cloud computing as there are many workflows with different QoS (Quality of Service) parameters. In this paper, we introduce a strategy, QoS based Workflow Scheduling (QWS) to schedule many workflows (Refer Figure 1) based on user given QoS parameters like Deadline, Reliability, and Cost etc. The strategy is to schedule the tasks based on QoS negotiation between user requirements and the services provided by Computation and Storage servers.*[2][3] *This architecture document describes the motivation, use and interaction of the three services that comprise version 3 of the UPnP-QoS Framework. While UPnP-QoS defines three services (listed above), it does not define a new device type. Since Quality of Service issues need to be solved for multiple usage scenarios, it is expected that vendors could use any UPnP device as a container for the services defined by UPnP-QoS.*

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: 1) 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. [5]

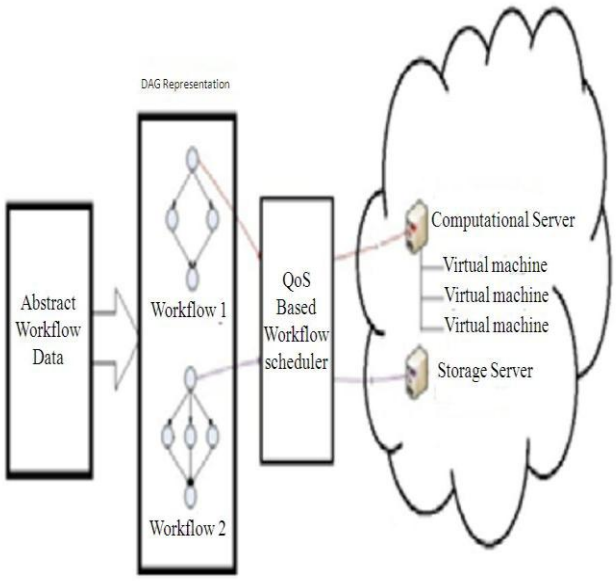


Figure 1 QoS Parameters

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 [6]. With the network traffic also increases, the quality of service (QoS) of multimedia applications degrades and finally completely falling down [9]. 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. Multimedia-aware cloud platforms are proposed to provide QoS for multimedia processing and storage [10] [12]. An IP multimedia subsystem with cloud computing architecture was proposed in to allow the users to access multimedia application via Android-based smart devices. The proposed design provides video streaming services with server virtualization technology [8].

Cloud computing has emerged as a global – infrastructure for applications by providing large scale services through the cloud servers. The services can be either storage service or computation service [1]. Hardware and software resources can be utilized by users as services. These services can be configured dynamically by making use of virtualization.

Cloud computing provides a computing environment for the applications which can be represented by the workflow. Workflow is a sequence of tasks processed in a specific order based on dependency of services between these tasks [9]. A workflow has a set of QoS parameters and it is

represented as a Directed Acyclic Graph (DAG) in which the nodes represent individual application tasks and directed arcs stand for precedence relationship among the tasks. Mapping between these tasks and services depends on the scheduling algorithm which is an NP complete problem [4].

Scheduling of workflows is a challenging one when many workflows are considered with many QoS parameters. There are many scheduling algorithms developed for QoS parameters which consider either Execution time or Budget constraints or both. Along with these parameters, reliability is also an important factor to be considered in various applications. For example in some real-time applications like medical surgery, banking, etc., require urgent execution of workflows. So workflow has to exhibit high levels of reliability because applications process may get delay due to workflow failures [15].

In this paper, we have considered deadline, reliability and cost as QoS parameters for scheduling. Each task should have some parameters to satisfy QoS requirements of a workflow, but in most of the cases, user will mention the QoS parameters for whole workflow. So in the proposed strategy, QoS based Workflow Scheduling (QWS), calculates the surplus information to achieve QoS negotiation for a workflow by using the distribution of parameters among tasks (Refer Figure 2). QWS accepts multiple workflows and multiple QoS parameters from the users at any time and it reduces make span and cost by considering reliability factor, which increases success rate of scheduling.

QWS ARCHITECTURE AND METHODOLOGY

System Model

The architecture of the QWS in a cloud environment. There are two major types of servers in cloud which are storage server and computational server. Storage server provides the service related to data storage and modification which does not require any mapping of services. Computational server provides the service related to computing resources which requires mapping of services based on QoS parameters required by a task. QWS process is designed by using three modules which are Preprocessor module (PM), Scheduler module (SM) and Executor module (EM) with rescheduling if required (Backfilling) [15].

Problem Definition

Workflow ωi is represented by a set of four tuples which are <Ti, j, Di, Ri, Ci>. Ti, j is a set of finite tasks {Ti, 1, Ti, 2, Ti, 3… Ti, j}. Each task Ti, j has a set of attributes like task-id, deadline, execution time, datasets and services needed, size, etc.

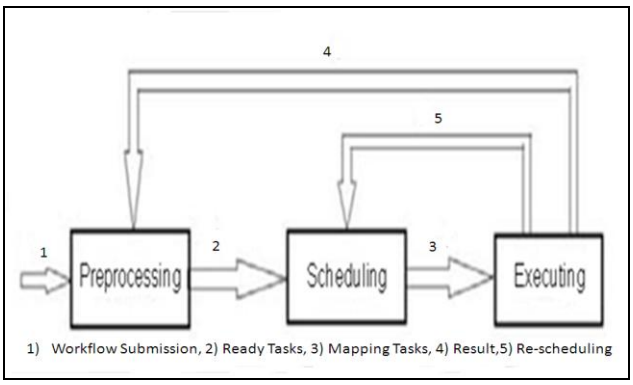


Figure 2 Control Flow Diagram

Architecture Summary

This section is a brief overview of the UPnP-QoS Architecture. UPnP-QoS defines three services [13]. These are the QoS Policy Holder Service, the QoS Manager Service and the QoS Device Service. The numbers in parenthesis indicate the order in which the UPnP-QoS actions are invoked [11]. To illustrate the relationships of various UPnP-QoS components, an example scenario with a simple sequence of setup steps is described below. Fundamentally, UPnP-QoS manages the QoS for a traffic stream that flows between a source and a sink device [13]. A traffic stream is viewed as a uni-directional flow from a source device to a sink device, possibly passing through intermediate devices. In the interaction described in this section, a Control Point application is assumed to have the knowledge of source, sink and content characteristics to be streamed, along with the content's Traffic Specification (TSPEC) [7].

The Control Point constructs a Traffic Descriptor structure and requests a QoS Manager Service on the UPnP network to setup QoS for traffic stream (Refer Figure 3). The Control Point in the QoS Manager Service (from hereon referred to as QoS Manager) requests the QoS Policy Holder Service to provide appropriate policy for the traffic stream described by the Traffic Descriptor structure. Based on this policy, the QoS Manager configures the QoS Device Service(s) for establishing the QoS for the new traffic stream [15].

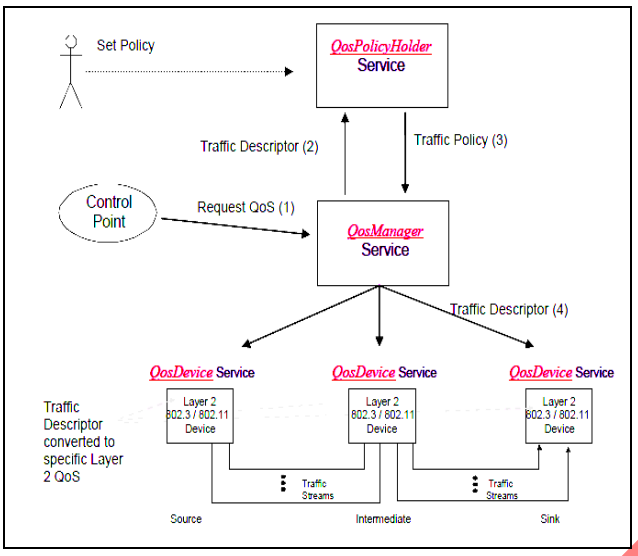


Figure 3 UPnP-QoS Architecture Overview

Policy-based QoS provides a way to assign priority for traffic streams and is also the basis for preemption. A policy-based QoS system allows an individual or entity to define rules, based on traffic characteristics and to manage the traffic's QoS in the context of the policy system. These rules are then applied to the characteristics of a request to determine the QoS applied. The rules utilize characteristics such as network address or application type [8]. The QoS Policy Holder Service provides the mechanism for classifying and ranking traffic streams according to information provided with the action to request QoS for a particular traffic stream. The type of information provided in the Traffic Descriptor structure includes, among other items, traffic class (best-effort, video, voice, etc.), the source and destination IP addresses for the stream, the TSPEC structure, application name, username, etc. The QoS Policy Holder Service examines the information provided in the Traffic Descriptor structure and returns the importance, in the form of a Traffic Policy structure [10].

Prioritized QoS

This section describes the interaction between the services for a request for prioritized QoS. Illustrates the interaction [8]. First the Control Point composes a request to the QoS Manager Service based, for instance, on the information in the Content Directory Service [CDS: 1]. Then the control point invokes the QM: Request Traffic QoS () action.

The QoS Manager collects information from the various QoS Device Services in the network. It obtains path information, with the QD: Get Path Information () action or other QoS related information with the QD: Get Extended QoS State () action.

The Traffic Importance Number is determined by the QoS Policy Holder Service and returned following the QPH: Get Traffic Policy () request. Based on the information available in the traffic descriptor (in particular Traffic Class) the QoS Policy Holder Service provides the Traffic Importance Number. In the case where the QoS Policy Holder Service cannot be used, Traffic Importance Number is determined by the QoS Manager using default policies [2].

The QoS Manager communicates with the QD: Setup Traffic QoS () and QD: Admit Traffic QoS () actions (Refer Figure 4) to the QoS Device Services with a Traffic Descriptor structure containing a Traffic Importance Number consistent with the QoS Policy[15]. The following diagram depicts the sequence of messaging for the setting up QoS for a given traffic stream.

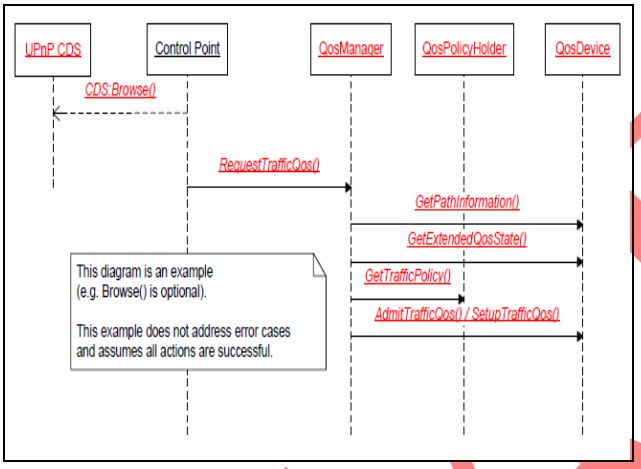


Figure 4 Interaction Diagram for Request Traffic QoS action for prioritized QoS setup

Interfaces and Links

Every QoS Device Service will have at least one interface which is defined as the point of interconnection between a device and a network (Refer Figure 5). Incoming or outgoing traffic streams use an interface to get from or to the network. "Wireless Network Connection". An interface is of a single technology such as Ethernet. An interface connects the device to the network and thereby to other devices. A link is an (possibly bidirectional) direct connection between two devices for data exchange. An interface can contain multiple links. In a device, a link can only belong to one interface. Different links can have different properties [10]. For example, in a wireless network the link from the Access Point to one station can have a different signal level and different throughput than the link from the same Access Point to another station. Within an Interface, links are identified through their Link Id [12].

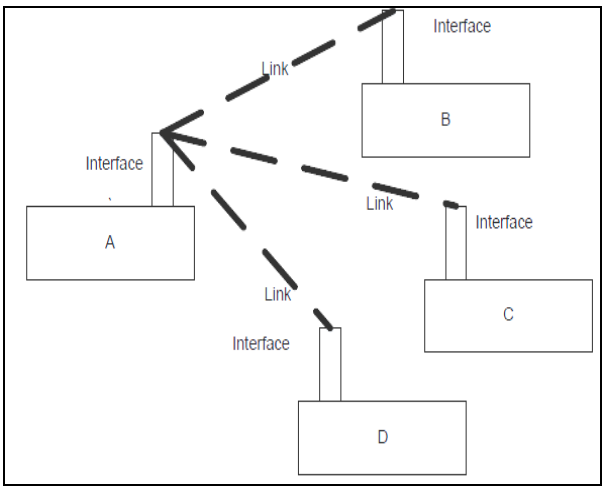


Figure 5 Interfaces and Links

Test Setup

First the possible correlation between latency and throughput is investigated, based on the results. 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. 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.

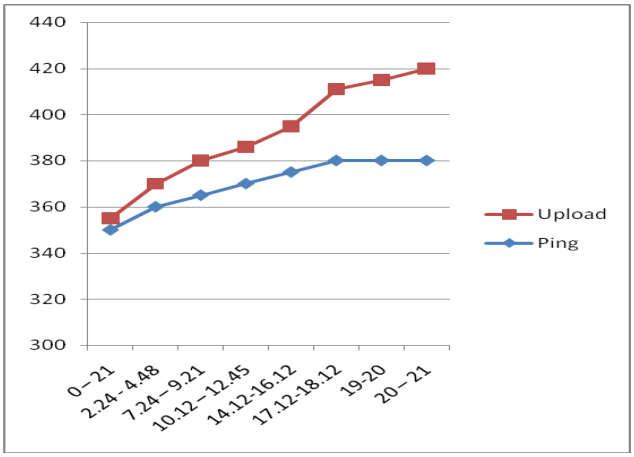
Seconds	Ping	Upload
0 – 21	350	5
2.24 - 4.48	360	10
7.24 – 9.21	365	15
10.12 – 12.45	370	16
14.12-16.12	375	20
17.12-18.12	380	31
19-20	380	35
20 – 21	380	40

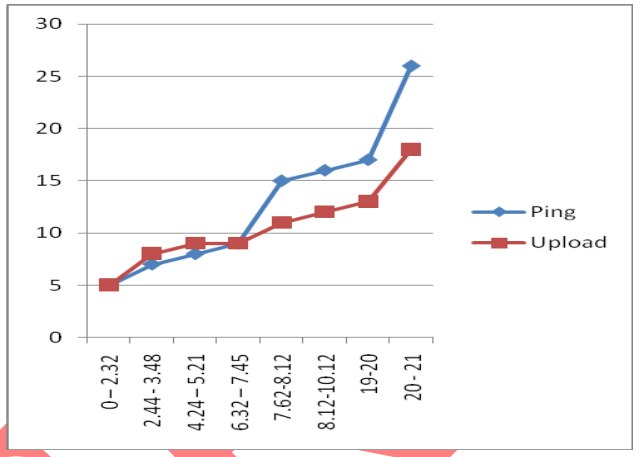
Seconds	Ping	Upload
0 – 21	350	5
2.24 - 3.48	355	10
5.24 – 9.21	365	15

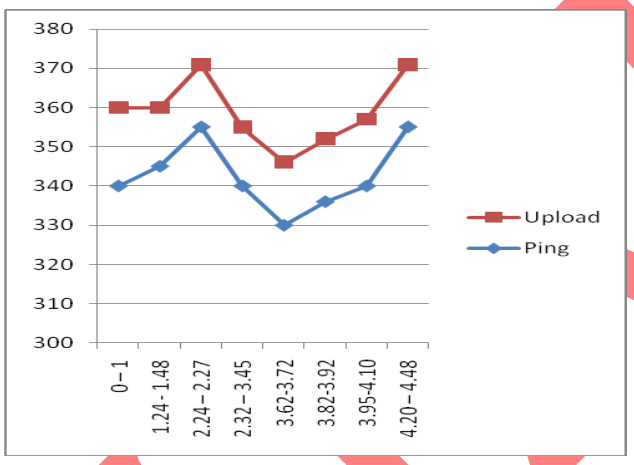
10.32 – 12.45	370	16
12.62-16.12	375	22
18.12-18.12	385	34
19-20	390	45
20 – 21	400	55

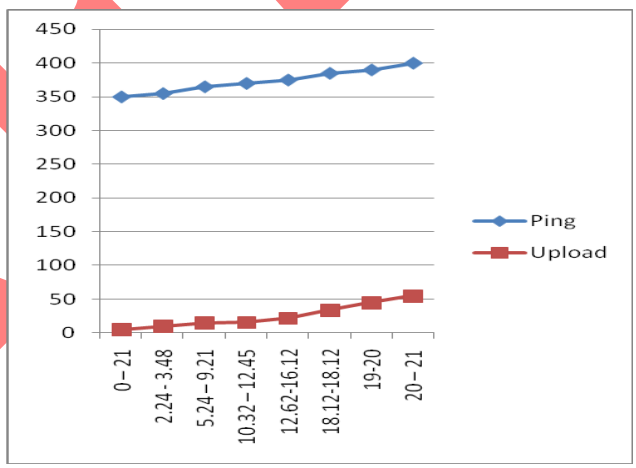
Seconds	Ping	Upload
0 – 2.32	5	5
2.44 - 3.48	7	8
4.24 – 5.21	8	9
6.32 – 7.45	9	9
7.62-8.12	15	11
8.12-10.12	16	12
19-20	17	13
20 - 21	26	18

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









CONCLUSION

The workflows in cloud computing platform have different QoS requirements. The main goal is to schedule many workflows by considering its QoS requirements. Many existing systems have addressed either for deadline or cost or both but not for reliability. The proposed algorithm, QoS based workflow scheduling (QWS), allows users to execute their workflows by satisfying their QoS requirements like deadline, cost and reliability.

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.

REFERENCE

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. 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. Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Breandic, "Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility", Future Generation Computer Systems, Elsevier Science, Amsterdam, June 2009, Volume 25, Number 6, pp. 599-616.
7. C. Germain-Renaud and O.Rana, "The Convergence of Clouds, Grids and Autonomics" IEEE Internet Computing, p. 9, 2009
8. Z. Shi and J.J.Donagarra, "Scheduling workflow applications on processors with different capabilities," Future Gen. Computer systems 22 (2006) 665-675
9. M.R. Garey and D.S. Johnson, "A Guide to the Theory of NP-Completeness", Computers and Intractability, New York, Freeman, 1979.
10. J. Yu and R. Buyya, "Workflow Scheduling Algorithms for Grid Computing", Metaheuristics for Scheduling in Distributed Computing Environments, F. X. a. A. Abraham, ed., Springer, 2008.
11. H. Topcuouglu, S. Hariri and M. Wu, "Performance-effective and low-complexity task scheduling for heterogeneous computing," IEEE Transactions on Parallel and Distribution Systems, vol. 13, no. 3, pp. 260-274, 2002.
12. 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.
13. 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

14. 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.

15. UPnP-QoS Architecture:3, For UPnP Version 1.0, Status: Standardized DCP, Date: November 30, 2008