Adama Science and Technology University

School of Graduate Studies

Department of Computing

SELECTIVE TESTING FOR CLOUD COMPUTING

GARAMU TILAHUN

Feb 2016

Adama Science and Technology University

School of Graduate Studies

Department of Computing

A Thesis Submitted to the Department of Computing of Adama Science and Technology University in Partial Fulfillment of the Requirements for Degree of Master of Science in Software Engineering

By

GARAMU TILAHUN

Feb 2016

Adama, Ethiopia

Adama Science and Technology University

School of Graduate Studies

Department of Computing

SELECTIVE TESTING FOR CLOUD COMPUTING

By

GARAMU TILAHUN

Names and Signature of Members of the Examining Board

Chair Person, Examining Board	Signature
FREZAWUD LEMA (PhD)	
Advisor	Signature
Examiner	Signature

ACKNOWLEDGMENT

First and foremost, I would like to thank to almighty God, who gave me strength to conduct this thesis and guided me throughout the whole process. No doubt without his blessing I am unable to complete my work.

I would like to express my deep gratitude to Dr. Frezewud Lemma, my research supervisor, for his patient and invaluable guidance throughout the research work. This study, in its current form, would not have been feasible without his guidance.

Moreover, I would like to thank Lemessa Aschala for configuring the required network for this study. I would like to give tribute to my family for their support and encouragement.
I would also like to thank my staff member Henock who give me his office and comfortable environment as well as computers which were necessary for laboratory test under the research.

Special thanks to all friends who gave me support and motivation by suggesting improvements in my work.

Table of Contents

Lists of Figures

Lists of Tables

Abbreviations and Acronyms

API=Application Program Interface

CC=Cloud Computing

CCP=Cloud Computing Provider

CID=Cloud Information Database

CP=Cloud Provider

CPU=Central Processing Unit

CRM=Customer Relationship Management

CSP=Cloud Service Provider

DNS=Domain Name System

DSD= Defense Signals Directorate

EBS=Amazon Elastic Block Storage

EC2=Elastic Compute Cloud

GNU=General Public License

HTTP=Hyper Text Transfer Protocol

IaaS=Infrastructure as a Service

IBM=International Business Machines

IP=Internet Protocol

ISO=International Organization for Standardization

JMT=Java Modeling Tools

KVM=Kernel Virtual Machine

LDAP: Lightweight Directory Access Protocol

MSS=Maximum Segment Size

MTU=Maximum Transmission Unit

NIST=National Institutes of Standards and Technology

NTP=Network Time Protocol

OCCI=Open cloud computing interface

OS=Operating System

PaaS=Platform as a Service

PBS=Public Broadcasting Service

PDAs=Personal Digital Assistance

Qos=Quality of service

RAM=Random Access Memory

RQs=Research Questions
SaaS=Software as a Service

SATA=Serial Advanced Technology Attachment

SQL=Structural Query Language

ST=Selective Testing

TCP=Transmission Control Protocol

UDP=User Datagram Protocol

URL=Uniform Resource Locator

VCPU=Virtual Central Processing Unit
VM=Virtual Machine

VMM=Virtual Machine Monitoring

VNC=Virtual Network Computing

Abstract

Cloud computing is the way in which a user acquire services such as software as a service (SaaS), platform as a service (PaaS) and infrastructure as a service (IaaS). Users get services by deploying their data and application on remotely servers. They pay only for the time the resources are acquired and based on computing resource running. They do not need to install and upgrade any software and hardware. Due to these benefits, organizations are willing to move their data into the cloud and minimize their overhead. In cloud computing, virtual machines play a vital role in providing services and deploying application for large number of users. Users should have information about a cloud performance before subscribing data and applications. The researcher overcomes problems by selective testing for cloud computing. The objective of this study is to give selective testing and to investigate cloud performance status and capacity of servers. Finding the parameters helps us to measure the actual performance of a cloud computing on some specific data and application. These parameters will help users, enterprises, developers and IT specialists to measure cloud performance based on their requirements and needs. In order to fulfill the objective of this study, an experiment and implementation had been done on lab test environment. Java programming language has been used for changing the algorithm as one method. Local cloud has been deployed using open source cloud which is known as openstack to conduct real test and Applications Manager tool used for testing services of Amazon EC2.

In the experiment, performance parameters (Throughput, response time and latency) including cost were tested and analyzed.

Keywords: *Cloud Computing, Performance evaluation, Performance measurements, Performance parameters, ST, Local cloud.*

CHAPTER ONE

1. Introduction

1.1. Background

Cloud computing is one of the greatest innovation in Information Technology in the recent decades and it is a design of technology ever made that provides services, applications and resources through a network [1].

Many several different companies and independent individuals have given definitions for cloud computing according to their strategies, visions and products. A working definition of Cloud Computing goes as cloud computing is "a model for allowing suitable, on-demand network access to a shared pool of configurable computing resources (i.e. networks, servers, storage and applications) that can be rapidly provisioned and released with minimal management effort or service provider contact." According to this definition, there are three types of service models, five major cloud characteristics and four classes of deployment models. Infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS) are service models of cloud computing. On-demand self-service, resource pool, elasticity and scalability, pay-per-use and broad-network access are the main characteristics of cloud computing. Depending on the kind of cloud deployment model, cloud computing divided in to public cloud, private cloud, hybrid cloud and community cloud delivery models [2].

There are various cloud computing providers for various aspects. Different providers give different service delivery models. For instance, in 2006, Amazon was the first cloud provider that provides infrastructure as a service. Openstack, Microsoft Azure, open nebula are also types of infrastructure service delivery models. Google mail and Salesforce are example for SaaS cloud providers. Red Hat OpenShiftF, Amazon SimpleDB/S3 and Google App Engine are examples for PaaS cloud vendors [3].

Cloud is beneficial for users, because cloud computing is virtualized technologies, has better service and collaboration, with wide availability of high bandwidth networks and high capacity of storage. This makes cloud to work in the web browser as a thin client that allows users to run any application. The resources might be computational power, has more storage, inter-networked, and run huge applications. Therefore, IT resources are elastic. They can be provisioned for the exact duration they are necessary and released when the consumer no longer needs them [4] [5] [6] Testing-as-a-Service delivers cloud servers quality management solutions in a flexible service model that accelerates the implementation of the quality center of excellence. IBM reported the experience on cloud testing small business division, resilience and cost-efficient cloud based development and testing environment is implemented [7] [8] [9].

The goal of testing is to determine how the system performs under different circumstances. These circumstances can be great load, resource scarcity, or specific types of data. It is also performed to evaluate different performance enhancements [11]. The researcher motivated to study on testing of cloud performance, because there are so many challenges and problems like throughput, response time, latency and cost of resource. Most of the time users blindly host their data and application in cloud. This faces users the risk of cloud computing performance.

The objectives of the study is selective testing for cloud performance to identify which and what factors affect service quality of cloud providers. To hit the objective of the study, we addressed solutions for problems using different approaches and algorithms.

Different proposed algorithms that had been taken as action were testing as service descriptions which were designed using object oriented programming language called class diagram. This had taken to implement basic ST interface called STFCC. The problem raised was solved using Java programming language. In addition to this methodology, local cloud lab test environment was deployed for real test data and application. Since the study focused on the selective testing (ST), it is indispensable to compare and evaluate several different cloud providers service. Applications Manager (AppManager) Tool was used to experiment and display performance of Amazon EC2.

All the steps and procedures answers for the problems mentioned. This indicates that the thesis raised

an idea that needs attention by users and providers as well as new comers in cloud providing. Because the algorithms and architectures defined and designed informs how to give selective testing and protect users from the risk of performance.

1.1. Statement of the problem and its justifications

At the present time selective testing is very common due to wide use of internet clouds and large number of applications provided on cloud computing. Conversely, despite the flow in activity and interest, there are significant, determined concerns about cloud computing that are impeding the momentum and will eventually compromise the vision of cloud computing as a new information technology achieving model [5].

There are so many cloud challenges in terms of performance: scalability, response time, latency or packet delay, throughput, etc. These are the main problem of cloud service providers in the provision of network performance for their users. So, measuring these factors is very important for any organizations before going to subscribe their data and application in remote server. Before subscribing applications into the cloud, it is recommended that one needs to flag areas of potential performance weakness, implement measures to prevent poor performance and perform ongoing performance measurement to find and fix problems as well as continuously determine where to host application. Client and server do have a clear goal, which allow the exchange of information over a communication network between two end points. When it comes to test networks, it is necessary to do it with the expected traffic that will go through the network whenever is used. Using different tools, designers can implement, simulate and evaluate their cloud application and use them in any real cloud environment without any code change.

Public IaaS providers ensure that an organization can rent significant external resources instead of purchasing its own. Types of cloud deployment model affect Small-to-medium-sized business and can considerably decrease their expected service. No capital investments for building an infrastructure, moderate ongoing costs, provisioning new computational power requires minimum effort, no staff dedicated entirely to maintain the hardware setup and high availability. In short, when the number of user's request increase, performance will be decreased. However, having one's own cloud will provide fast response time, throughput and less packet loss.

The other problem of cloud service provider's performance degradation is pushing and pulling computing resource from single or one machine (i.e. CPU, RAM, Storage, etc.). This is very crucial concept in case of best or bad provision of performance. Most probably, cloud providers are business oriented. Lack of cloud management and configuration or lack of resource without attention, degrades leveraging good performance for consumers. When such things exist, even availability of service becomes outage. In middle of this, the consumers face the problem of performance on their business. When we use traditional server farm, more cost is invested on buying the device or suddenly the server may be down and stop to provide function. Since cloud computing has a potential to overcome such challenges and problems, most organizations are keen to host their software in the cloud. Since cloud computing is elastic, on-demand service and has distributed data center, most of the risk of traditional server farm become solved. But there are also drawbacks of cloud computing, which need selective testing to test providers performance. The drawbacks of cloud computing challenges those users who are going to deploy services remotely.

Location of server has problem on performance, because distance is inversely proportional to response time. Simultaneously requesting service of cloud also has problem on request time. When number data source request increases to use cloud, the server becomes bottlenecked and the requested data take much time to get response from the cloud. The server becomes stressed and there may be long processing time to fetch data and performance of user or even it can be one enterprise's access will be decreased.

Performance reduction in cloud computing could happen; even type of hypervisor used has influence. It is better to identify types of virtual machine monitoring tool that increase performance of cloud computing since hypervisor has its own impact on performance. One of the main features that differentiate cloud computing from the traditional computing is that clouds are usually programmable. A cloud system can run a number of machines in a complete isolation from each other and in millions of requests will come at the same time. As a result, applications can be easily provisioned only with the resources they require and later on destroyed if not necessary by using virtual machine so that the resources can be used for other purposes of applications. Better utilization is an advantage that comes from the virtualization technologies but cloud platforms provide mechanisms to control many physical nodes from a centralized point and thus organizing the nodes into server farms improves the process of their management.

As usual users are not familiar with using cloud computing. Even when they need to deploy their application, there is shortage of know how about the detail of cloud vendors. In this thesis, the decision support system is developed. This means giving more detail information about cloud providers system especially regarding their performance delivery. It is a familiar limitation that there is no standard shared systems that compare the performance or price against different cloud providers and give information of all providers service delivery.

The previous researcher had been studied performance of cloud by either deploying cloud itself or test using simulator tools. Their method does not satisfy and give full confidence for users. The problem raised will be solved through developing selective testing interface for cloud providers and deployed prototype of cloud service as a case study. Finally we analyzed performance result of cloud services.

1.2. Research question

The aim of this study has been to found solutions for the above-mentioned challenges. Therefore, the algorithm and approaches developed got answer for the following research questions.

- Can selective testing of cloud computing be an answer for minimizing the above mentioned problems?
- How to know the best cloud service providers and the specific characteristics of their performance?
- Is the performance of cloud provider services satisfactory for subscribing data and applications?

1.3. Objectives of the study

1.3.1. General objectives

The general objective of this study is selective testing of cloud performance with the main goal of finding the server limits, build a basic foundation and prove its feasibility.

1.3.2. Specific objectives

- To design selective testing of cloud performance architecture, this is applicable both to user and cloud service providers as a reference model.
- Design interface which accept customer business requirement and formulate testing as a service.
- To ground prototype using open source cloud platform.
- To determine performance status of cloud servers using request and response process.
- To analyze performance metrics such as throughput, latency or packet loss, response time, scalability of the server and finally get computing resource cost.
- To identify the limitations of cloud performance.

1.4. Scope and limitations of the study

Selective testing system was performed sample test using open source cloud called openstack and Application Manager Tool to bring sample performance data of Amazon cloud. It is focused on finding out cloud performance; therefore, the scope of this selective testing is to carry out and evaluate throughput, response time, scalability, latency and cost. In general the study was bounded by the following mentioned frontiers:

- Develop algorithm and approaches including selective testing interface
- Deploy local cloud on the laboratory environment

As the limitation, it is impossible to exceed one open source cloud to deploy on real environment. As usual, cloud is so complex to test by deploying on physical environment and the time given for the study restricts to test all providers.

1.5. Methodology of the study

To attain the objectives of this research the following methods and techniques has been used. We represent view of methodology that has been taken by using graph below.

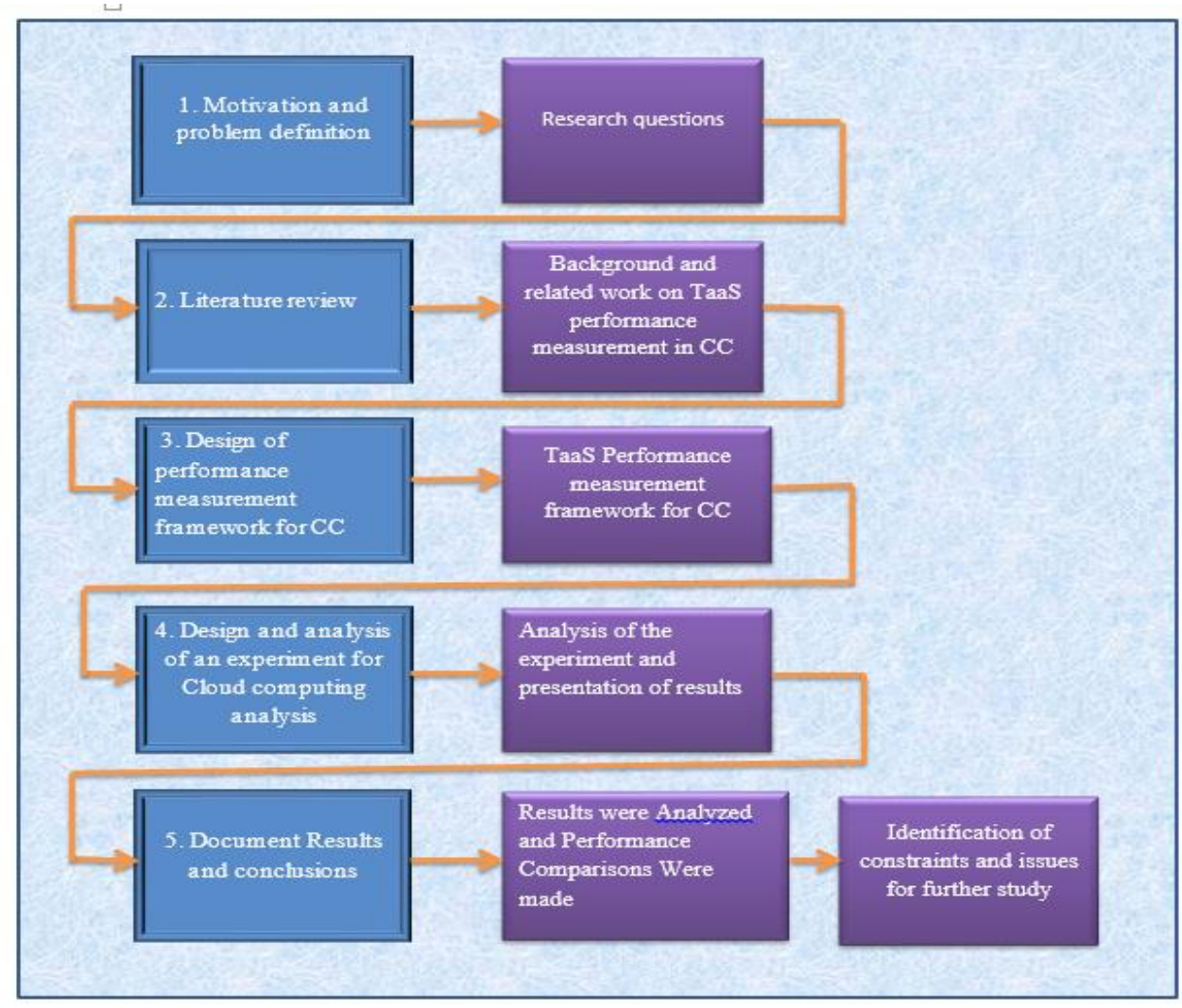

Figure 1.1 Graphical representation of research methodology

Literature review

Several previously proposed related literatures (books, articles and the Internet) are reviewed in order to have detail understanding on this research works. Different techniques and tools, which are relevant for the current research are analyzed, modified and used from previous work mechanisms.

1.5.1. Tools needed for the thesis work

Operating system

We believe that, for the development of this thesis, Linux would be more suitable. Linux is a computer OS created by developers and employees of many companies and organizations from every parts of the world in order to make this product free software. The main advantages of Linux are stability, performance and network functionality [15]. In this research, cloud platform used is openstack which is supported by Linux operating system

Front-end tool

Programming language

Several programming languages can be used for cloud programming. Java programming language is one of the most important languages which support this research work. Java Eclipse is selected since it can go with our work and also it is the latest version of java software. Using this language, the algorithm have been written and made selective testing interface. This could simplify choosing cloud vendors' status and provide deep information about cloud computing services.

Back-end tool

Oracle database

We can store our data in the database. For example oracle is one of database tool which store large number of files. The researcher do not select oracle database since the researcher is not familiar with oracle.

MySQL database

The researcher select MySQL database since familiar with it. Information of cloud providers stored in the database called cloud information database (CID). When the users need to identify cloud

providers, data or information retrieved from CID. It is an open source tool which can be used to store content and configuration information. Additionally it is a relational database management system which is frequently chosen by developers, because it provides speedy website loading, reliability, and ease of use [16].

Cloud management software

Openstack is a cloud infrastructure platform that controls large pools of compute, storage and networking resources. All nodes are managed through a horizon dashboard that gives administrators control while empowering users to provision resources through a web interface [17]. It can also be defined as compute, networking and storage that provide a pool of services like CPU, memory and storage. It is made available to consumers as building box through applications. It was first released by the cooperation of RAKSPACE and NASA in 2010 [18] [19] [20].

Xen hypervisor

Several Virtual Machine Managers (VMMs) have been developed and are currently used in cloud computing environments. Xen hypervisor is used in this thesis work. It is situated between the hardware and operating systems. It is responsible for CPU schedules and memory partitioning of various VMs running on the hardware device. The hypervisor is not only abstracts the hardware for the VMs but also controls the execution of VMs as they share the common processing environment. Xen implements Para virtualization as well as hardware-assisted virtualization [21] [22]. It is licensed under the GNU General Public License (GPLv2).

1.6. Significance of the study

The benefits of cloud performance testing and selective testing is very important to overcome some challenges of cloud computing. Testing cloud computing forces to reduce the risk of service availability, response time, throughput and packet delay while increasing testing effectiveness. The non-cost factors include utility like on-demand flexibility, enhanced collaboration, and greater level of efficiency and, most importantly reduced time-to- market for key business applications. It can provide full confidence to deploy the service in the cloud because it is possible to measure and evaluate what is going on and happen on the cloud server. The other benefit of this research is to reduce concern about infrastructure for those who are to deploy data and application in cloud. This would be happen because of resource virtualization. Any users or enterprises can be the beneficiary of this research's result. Results of this study may be used as model reference for the provider as well as the user. For the cloud service provider, it suggests them to give more attention and to prepare their services much better. For the user, it uses as references before they blindly give their data and application in hands of third party.

Beneficiaries of the research

- The organization who need to deploy cloud
- Any enterprises that are going to subscribe their data and application in the cloud.
- Governmental and non-governmental institutions and individuals who need to share and users of cloud computing, etc.

Thesis organizations

The study contains totally five chapters and organized as follows.

Chapter one: - Discusses an introduction about the study giving highlight about the area on which the study focuses. ST for cloud performance in cloud computing discussed well. Problems of the statement, objective of the study, significance, scope, and methodology used were described in detail.

Chapter 2:- This chapter is devoted to literature review and related works. More definition and explanation of CC and ST discussed were Perception of cloud computing, characteristics of CC, advantage of testing were deliberated. Furthermore, descriptions such as testing as a service,

relating performance analysis of different cloud provider have come across. In addition, related works in the area of cloud computing have been discussed in detail.

Chapter 3:- In this chapter, we defined the testing as service architecture and proposed algorithms for the study. This chapter directly faces with the contribution and how to solve research questions raised so far. Different mechanisms and techniques have been modeled with at least three basic approaches.

Chapter 4:- Is about experiments, implementations and results. Algorithm designed had been solved by methods of deploying local cloud and by providing prototype on the lab test.

Chapter 5:- This chapter deals with discussions, conclusions and future directions. Based on the result obtained from the study, the research concluded by reporting findings and put recommendations for future works.

CHAPTER TWO

2. Literature review and related works

2.1. Literature review

A literature review is used to find out the most relevant updated research articles to better understand, evaluate, analyze and compare the desired research area. The main reason for literature review is to find and summarize the existing information about cloud performance. The review should be carrying out in order to find the following activities.

- To find out the existing practical evidence of the cloud performance.
- To find out the gaps of cloud environment and its performance to further investigate solutions for the problem.
- To suggest and propose a new model that assisted this research.

2.2. Overview of cloud computing

Experts in the cloud computing industry and providers give their own definition for the term cloud computing. Indirectly the term used everywhere when we used internet but it is not known by the consumers. Nowadays there is not yet consensus for what the term means. Evaluating some of the existing definitions helps to clarify the term and what it involves (or might involve). Here are some of the definitions quoted from different experts and companies:

> "Cloud computing enables convenient network access to a shared pool of configurable computing resources. This can be provisioned and launched with the minimum management effort or service provider interaction. This can be defined in plain terms is the capacity to reach convenient infrastructures for end users to utilize parts of bulk resources and the resources can be acquired quickly and easily. Due to the nature of infrastructure, application and softwares being saved 'in the cloud which enables access from any fixed and movable devices like mobile, which can link up to the internet" [1] [2].

Cloud computing is defined as a concept, where system users save the information on remote servers which is controlled and managed the applications which stored inside the server and accessed from other remote locations [3].

"A cloud computing is a class of parallel and distributed distantly containing of a combined networked system and virtualized computers that are dynamically released and presented as one or more unified computing resources based on SLAs. This is based on established negotiation between the service provider and consumers" [4].

Cloud Computing is a way of launching the internet to the software of consumers or other IT services on demand. Users share infrastructures such as processing power, storage space, bandwidth, memory, and software. They pay as they go and use what they need at any given time, keeping cost to the user down. This process indicates that cloud Computing is business-oriented model [5].

2.3. Cloud computing architecture

Cloud computing architecture is defined by NIST as it has five essential characteristics, three cloud services models and four cloud deployment models [1].

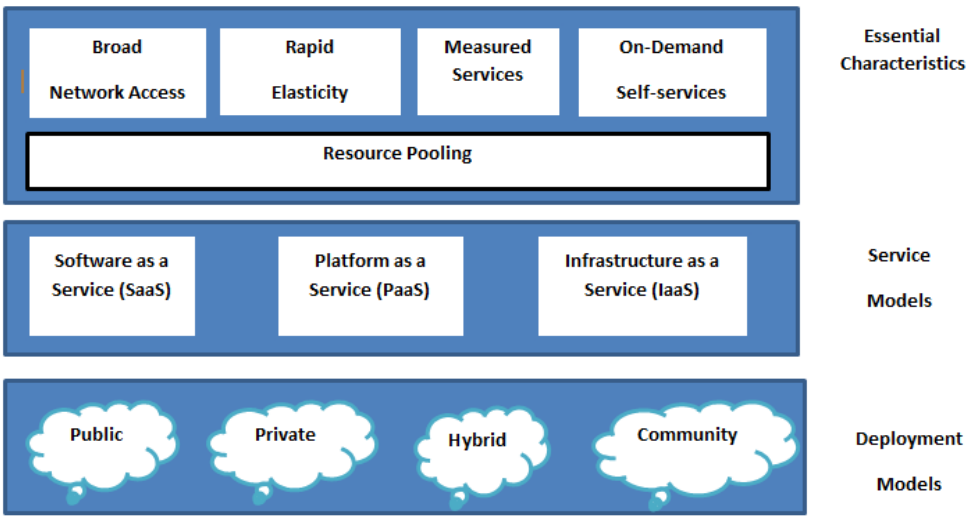

Figure 2-1 Visual model of NIST working definition of cloud computing (adapted from [6])

2.3.1. Essential characteristics of cloud computing

A basic condition that a cloud provider must fulfill is the ability to deliver computing resources whenever customers need them. From customer side, the available computing resources are nearly infinite. This means the customers are not limited to the set of servers located at one site and it is the responsibility of the cloud computing provider to have sufficient resources to satisfy the requirements of all their customers. The recognized essential characteristics of a cloud computing environment are discussed as follow.

Rapid elasticity

Rapid elasticity is well-defined as the ability to scale resources both up and down as needed. The provision and release of the resources, instantly and elastically, are preferably done in a programmed way in order to allow a consumer to quick scale out and in which is considered as rapid elasticity [7].

Measured service

Measured service refers; cloud provider is checking and pointing all aspects of the cloud service. Usage of resource is controlled and optimized by cloud systems automatically. It has been done by leveraging a metering capability at some level of abstraction appropriate to the type of service (i.e., storage, processing, bandwidth, and active user account.

On-Demand self service

The on-demand and self-service aspect means that a consumer of cloud computing can unilaterally deliver computing abilities such as server time and network storage, as needed dynamically without requiring human involvement with each service provider. This enables the customers to avoid making unnecessary upfront investment of servers and computing resources.

Broad network access

This means that the cloud provider's skills are accessible over the network and can be accessed through standard mechanisms that stimulate use by varied thin-client or thick-client platforms (e.g., mobile phones, laptops, and PDAs).

Resource pooling

With the resource pooling, the provider's computing resources are shared to serve multiple consumers using a multi-tenant model, with dissimilar physical and virtual resources dynamically assigned and reassigned according to user demand.

Based on the definitions of National Institute of Standards and Technology Laboratory, he service model can be classified into three types. These are:

- Software as a Service (SaaS)
- Platform as a Service (PaaS) and
- Infrastructure as a Service (IaaS).

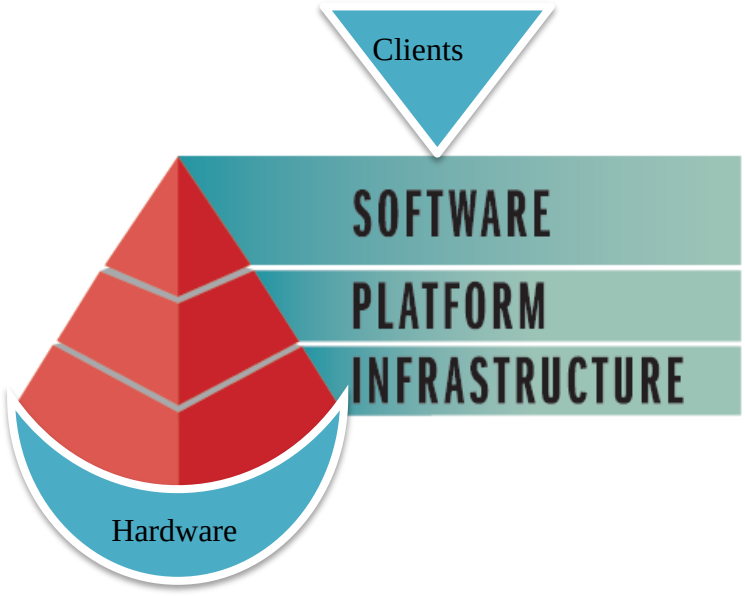

Figure 2-2 Cloud computing service delivery stack: adapted from ([8])

Software-as-a-Service (SaaS)

We can define SaaS as software that is deployed over the internet. Anybody can access the applications through a thin client interface such as a Web-based e-mail. The consumers have no authority to control over the infrastructure, operating system (OS), network, storage and even the service they use and gain. Salesforce and Google are a typical example for SaaS [1].

Platform-as-a-Service (PaaS)

PaaS platform allows the creation of web applications quickly and easily and it has no complexity of buying and maintaining the software and infrastructure underneath it. This can been done by using libraries and tools provided online. The ability delivered to the consumer is to deploy onto the cloud infrastructure. PaaS supports cost-efficient deployment of applications without complexity of buying and managing the underlying hardware and software layers. Even though the consumers have no control over the infrastructure, they have full control on the deployed application. Microsoft azure, Amazon SimpleDB/S3 and Google App Engine are examples for PaaS.

Infrastructure-as-a-Service (IaaS)

Infrastructure as a Service (IaaS) specifies to offerings of necessary computer resource infrastructure as a service (IaaS) providers which are full control of virtual resources. Amazon EC2, GoGrid, FlexiScale is some examples of IaaS service model cloud computing.

2.3.2. Cloud computing deployment models

In cloud deployment model, one size doesn't fit all deployment. Different enterprises and sectors have different requirements and concerns. According to NIST, the deployment model can be classified into four as discussed below [9].

Public Cloud

In the public cloud model, the cloud infrastructure is provided by a Cloud Service Provider (CSP) for a number of customers. This means cloud infrastructure is available to the general public or a large industry group and it is possessed by an organization marketing cloud services. In simple terms, public cloud services are characterized as being available to clients from a third-party service provider through the internet. The term "public" does not always means free even if it may be free or fairly inexpensive to use. A public cloud does not mean that user's data is publicly visible instead providers classically provide an access control mechanism for their users [7].

Private cloud

The cloud infrastructure is provided by a single organization for its different business units. The infrastructure may exist on or off-premises, and is maintained by the organization or third parties in private cloud. The cloud infrastructure operates only for the organization [1].

Hybrid cloud

In the hybrid cloud model, the cloud infrastructure offers extension for the private cloud to pool local resources with resources of the public cloud. Hybrid cloud is combination of both private and public cloud [10].

Community cloud

In the case of community cloud, the cloud infrastructure is pooled among customers with similar requirements such as security policies, common mission, requirements and compliance issues. The management and controlling of a cloud environment can be handled by third parties. Some parts of cloud computing infrastructure are to be deployed for industry, commercial or academic purposes. Many types of cloud computing architecture have been developed as public or private cloud [7].

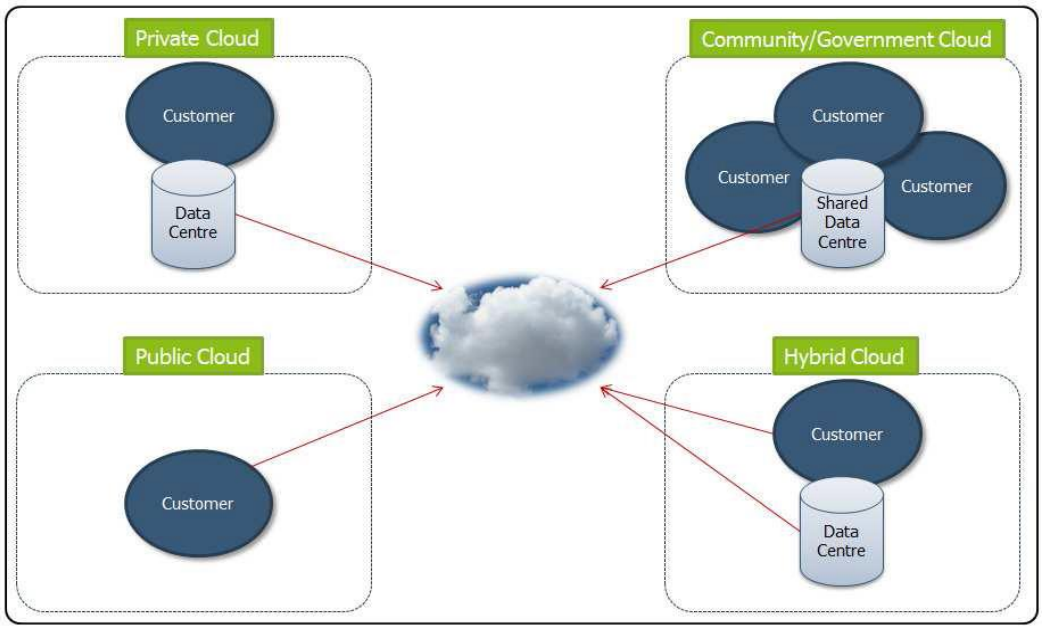

Figure 2-3 Conceptual deployments models of cloud computing (adapted from)

Benefits of cloud computing

More than a semantic dispute around classification, we believe that in order to maximize the benefits that cloud computing delivers, a solution needs to demonstrate these particular characteristics of cloud computing. These were discussed under the section of essential characteristics of CC [11].

- Cloud computing can greatly benefit public sector organizations of all types and sizes by:

Reducing costs

Today most of users' applications are being moved to CC to use resource wisely by minimized cost. Costs can certainly be reduced. For instance, you don't have to invest in servers at your site, because somebody else is doing that for you. And, of course there's management. User finds only the hardware required for each project. This potentially reduces the risk for institutions which want to build an elastic system, thus providing greater flexibility since the user is only paying for the required infrastructure while reserving the option to increase up services as needed in the future.

Capacity, scalability and speed

There are three additional key benefits of the cloud depending on its ability to deploy, scalability, and speed. A cloud subscriber no longer has to buy more computing capacity than what is needed at any given time since the cloud is scalable. Clouds are designed to distribute as much computing power as any user needs. The cloud resources are projected to simplify the developer's dependence on any specific hardware. Just like the utility of electricity, the cloud can stretch or shrink according to the varying needs of individual subscribers [11].

Backup and recovery

The procedure of back up and recovering data is simplified since it resided on the cloud and not on a physical device. Several providers of cloud offer reliable and flexible backup or recovery solutions. In some cases, the cloud itself is used solely for the sake of backup repository of the data located in local computers. Public cloud computing is built on a backbone of data security which is combined with geographic redundancy for maximum availability [11].

Availability, geography and mobility

Cloud computing is ubiquitous. Cloud technology endeavor access, through the internet, to more data stored in the cloud at any time and from any location in other words, anything, at anytime and anywhere.

Increased storage capacity

Cloud computing hold and store more data compared to a local computer and in a way it offers almost unlimited storage capacity. It eradicates concerns about running out of storage space and at the same time it spares businesses the need to modify their computer hardware, more to reducing the overall IT costs.

2.4. Understanding selective testing for cloud computing performance concept

The desirable characteristics of the target selective testing for cloud computing, test execution environment are based on the CPs and users' requirements. Before subscribing the application in the cloud, testing cloud service whether service providers address users' expectation is the crucial thing to understand. The understanding of users must go with what things can be tested about the cloud. The users look for a service depending on their requirement, service level agreement and some other underlined matters. Sometimes the bargain point between user and cloud service providers fails to meet user's expectation. So, the users should have information about the cloud's throughput, response time, latency, scalability and cost. This could be executed when the users request increase, after the load is maximized or the server stressed [12] [13].

In cloud computing, there are challenges and risks related to performance and service delivery. To overcome the problem which affects users of cloud service, testing the cloud performance is crucial. When there is pre-checking different cloud providers' performance through the mechanism of compare and contrast each and every service delivery, we can definitely decide their performance quality and pledge our application or data in cloud.

2.4.1. Purposes of performance testing

Performance testing has input value for the users which is important to consider the adequacy of the cloud's performance. When the cloud tester gives services of selective testing for users of cloud, different service elements that evaluate cloud computing capacity will be measured and assessed. For example, when the system is operating under realistic loads, response time can be satisfactory or not. We do not know whether cloud systems can accommodate the estimated demand. Large number of users may access the cloud simultaneously and the system may not be operating correctly. When consumers of cloud service added, the performance degradation has to be considered (i.e. does the system performs well under heavy and peak loads?) Cloud performance testing accommodates so many things of performance evaluation parameters. Listing these all things is very difficult but as much a possible the mechanism of selective testing solve many challenges for users, since the method is pre checking the proficiency of cloud provider services.

2.5. Cloud computing pricing policy

Cloud providers' offer different services according to cloud services they give. They decide cost based on the instance type, amount of data transfer (i.e. downloading and uploading) and based on location of data center. For example, the lowest VM instance size is called micro, tiny or small. There is also medium size VM instance (m1.medium) and highest performance VM instance known as large. Therefore users ought to choose advisable value for element variables to decrease their expense. This can be decided after the consumer determines deployment scenario which means their data size [14] [15].

2.6. Actors and roles in the cloud computing value network

The NIST CC reference design architecture identified five major actors: cloud consumer, cloud provider, cloud auditor, cloud broker, and cloud carrier.

Table 2-1 Actors in cloud computing (adapted from [1])

No.	Actor	Definition
1.	Cloud Consumer	Person or any organization that maintains a business relationship with, and uses service from, Cloud Providers (CPs).
2.	Cloud Provider	Person or company, or entity responsible for making a service available to cloud consumers (CCs).
3.	Cloud Auditor	A party that can foreseeable independent assessment of cloud services, information system processes and performance of the cloud implementation.
4.	Cloud Broker	An entity which manages the use, performance, and delivery of cloud services, and negotiates relationships between CPs and CCs.
5.	Cloud Carrier	The midway that provides connectivity and transport of cloud services from CPs to CCs.

2.7. Cloud computing providers

Today cloud service providers are flourishing. These providers can be categorized as commercial cloud service providers and non-commercial cloud service providers.

2.7.1. Commercial cloud computing providers

Amazon Web Services (AWS) cloud provider

The AWS is a commercial web service offering infrastructure. It is one of the prominent companies in the field of cloud computing. Amazon announced Amazon's Elastic Compute Cloud (EC2) [16] which deliver services via Amazon Web Services (AWS) and reflected mainly as an IaaS layer. AWS allows the users to manipulate the VMs sitting and configuration with a highly scalable platform. Amazon cloud provides a PaaS layer wherever the user can develop and deploy their applications which assure the customer's trust. Moreover, AWS provides a platform to deal with the database, SimpleDB [17].

AWS offers infrastructure as a service (IaaS), which includes:

- **Servers:** - This can be provided through Amazon Elastic Compute Cloud (Amazon EC2) which is one of the AWS.
- **Storage:** - Amazon Simple Storage Service (Amazon S3) is a web interface. It provides a storing service which can store and retrieve stored data on the web.
- **Operating Systems:** - AWS permits the user to choose from a wide variety of available operating systems already built in AWS. In addition, it allows the user to use different OS for one application.
- **Networking:** - This is applicable for different services which use different networking services such as Amazon Virtual Private Cloud (Amazon VPC) which uses a Virtual Private Network (VPN) connection.

Windows Azure cloud provider

Windows Azure platform provides friendly interfaces to deal with the heart of PaaS in developing and deploying different NET applications to distribute relational databases. Windows Azure gives service like SQL Azure which presents DB services. SQL Azure provides reporting, querying, and data synchronizing which contain relational databases. Features offered by Windows Azure include computing resources, storage, database, Virtual Machines (VMs), access control, caching, virtual network, service bus and business intelligence [18] [19].

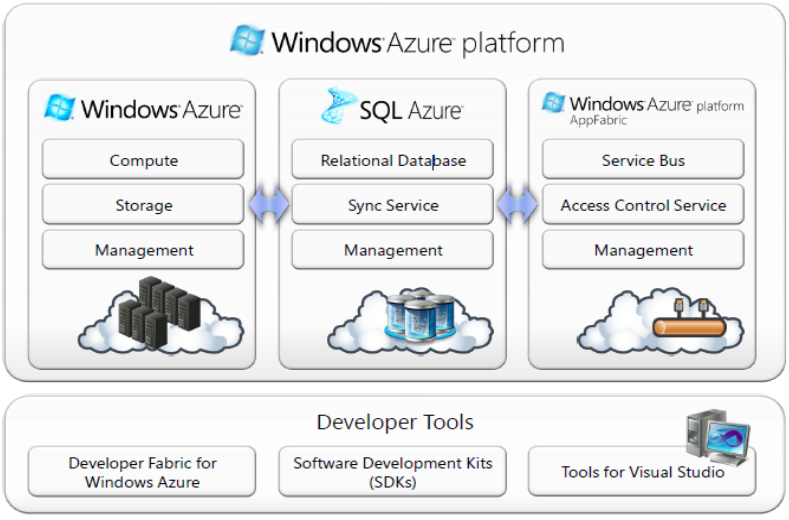

Figure 2-4 Windows Azure platform architecture (adapted from [18])

2.7.2. Open source cloud computing providers

Openstack

Openstack is open source software tools for building and managing cloud computing platforms for public and private clouds as well as it is software platform which provides an Infrastructure as a Service (IaaS) solution via interrelated services. In July 2010, NASA and Rackspace hosting launched open source cloud software known as OpenStack [20] [21].

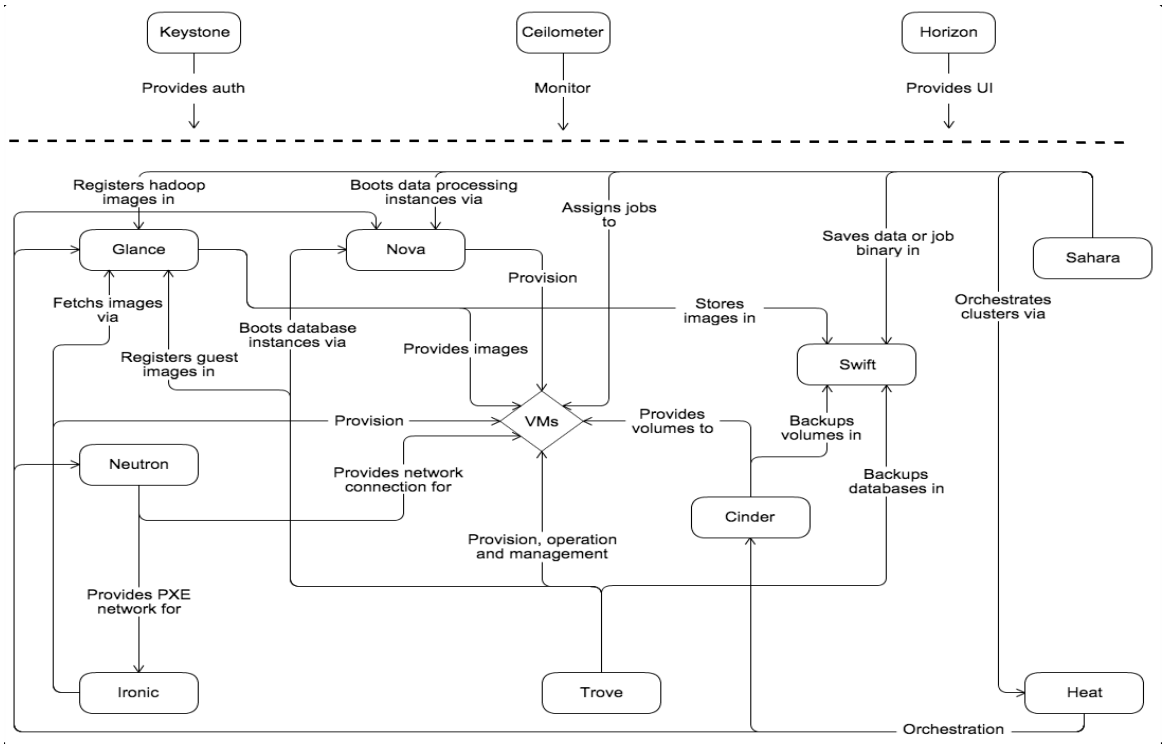

Figure 2-5 Basic Conceptual architecture of Openstack (adapted from [22])

Components of openstack

The OpenStack components have a significant impact on the overall design. OpenStack design architecture uses the following components [23] [24].

Compute (Nova): - OpenStack compute (Nova) is a cloud computing machine that used for deploying and managing large number of virtual machines and other instances to handle computing tasks.

Object Storage (Swift): - OpenStack object storage or swift is a scalable redundant storage system for objects and files. Objects and files are written to a number of disk drives spread on different servers of the data center.

Block Storage (Cinder): - It is called cinder which functions as a block storage component and able to access specific locations on a disk drive. Cinder provides consistent block level storage devices for use with OpenStack compute instances. In the case of Openstack, block storage system (Cinder) controls the creation, attaching, detaching of the block devices to servers.

Networking (Neutron): - Neutron provides the networking ability for OpenStack and it is a system for managing networks and IP addresses easily and efficiently.

Dashboard (Horizon): - This is the interface for OpenStack which provides administrators and users a graphical interface to access cloud based resources.

Identity Service (Keystone): - Keystone furnish identity services for OpenStack or it is a central directory of users mapped to the OpenStack services which they can access and gives authentication system across the cloud operating system and can integrate with existing backend directory services like LDAP.

Image Service (Glance): - It provides flavor of image services to OpenStack, discovery, registration and delivery services for disk and server images. It also allows images to be used as templates when deploying new instances.

Telemetry (Ceilometer): -Ceilometer gives telemetry services which allow the cloud to provide billing services to individual users of the cloud.

Orchestration (Heat): - It allows developers to store the requirements of a cloud application in a file that defines what resources are necessary for that application.

Database (Trove): - Trove provides database as a service which provides relational and non-relational database engines.

Eucalyptus

Eucalyptus is short expression of Elastic Utility Computing Architecture for Linking Program to Useful System. It was developed by university of california-santa Barbara for cloud computing to implement infrastructure as a service. In 2008, Eucalyptus become the first open source software which is compatible with AWS API for deploying on premise private cloud. Eucalyptus also provides an EC2 and S3 storage cloud computing platform compatibility. Thus, its services are available through EC2/S3 compatible APIs [25].

Eucalyptus has five high level components. These are:

Cloud Controller (CLC): - Cloud controller is the entry socket into the private cloud for end users. It also helps in managing virtualized resources.

Walrus: - We can implement bucket-based storage using walrus which is available on the cloud system.

Cluster Controller (CC): - It performs on a machine which has network connectivity that is running on node.

Controller and Cloud Controller: - Network and VMs managed through this node. All node controllers related with a single cloud computing must be in the same subnet.

Storage Controller (SC): The storage controller gives block-level network storage including support for Amazon Elastic Block Storage (EBS) and Node Controller (NC). It is installed on every compute node to manage VM activities with the execution, inspection, and termination of VM instances.

2.8. Virtualization and virtual machine management

Virtual machine (VM) concept was developed first by IBM in the 1960s to provide concurrent, interactive access to a mainframe computer. Virtualization is an essential technological characteristic of clouds which hide the technological difficulty from the users and enables enhanced flexibility (through aggregation, routing and translation). The virtual machine management is the process of creating, listing, migrating (both live and cold migration) and deleting virtual machines from the cloud management software. The reason for choosing this feature is that it allows the cloud consumers to choose software that manage the virtual machines (VM).

The virtualization paradigm is a good fit to cloud computing since it can improve resource utilization [26]. By using virtualized resources, cloud computing works with the same set of physical resources with a much broader user base.

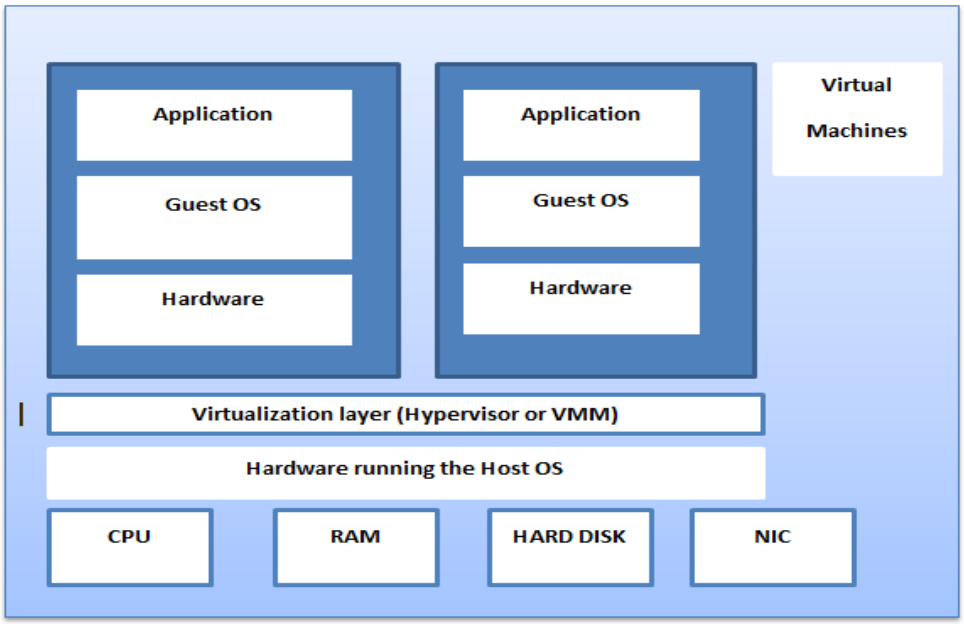

Figure 2-6 Virtualization technology (adapted from [14])

Table 2.2 shows the virtual machine management and characteristics provided by different cloud management software. 'Y' indicates that the feature is provided. 'N' indicates that the feature is not supported by the cloud management software. 'G' indicates that the cloud consumers can use the graphical interface to manage the virtual machine, and 'C' indicates that the cloud consumers can use the command-line interface to manage the virtual machine. In customization option, whether it is possible to create the virtual machine with different sizes (number of CPU cores and RAM size) is discussed. 'Y' indicates that the cloud consumers can create the virtual machine with different sizes and 'N' indicates that the cloud consumers can only create the virtual machine with a default size and at the last language written in descried.

Table 2-2 Cloud Platform and their Properties

Type	Cloud management software	Backend Hypervisor(s)	Virtual machine management	Graphical interface (G) / Command-line interface (C)	Customization option	Language written in
Commercial	Amazon EC2	Xen, KVM	Y	G+C	Y	Java PHP Python Ruby
	Microsoft Windows Azure	Xen, KVM	Y	G+C	Y	Java PHP Python Ruby
Open-source	Openstack	Xen, KVM and VMware ESX	Y	G+C	Y	Python, shell scripts
	Eucalyptus	Xen and KVM	Y	C	N	Java, C/C++, Python, Perl, Shell scripts

2.9. Cloud Interface

The cloud interface is a way to administer the virtual machine images, virtual networks. It is a method of access to and utilization of a cloud storage system. If cloud users have an experience with a specific cloud interface (e.g., Amazon / Libvirt), they can use the known cloud interface to interact with the software instead of learning a new interface. Table 2.3 shows the list of cloud interfaces provided by the software such as proprietary API. The proprietary API is specific to cloud management software and it cannot be used by any other cloud management software.

Table 2-3 Cloud interface provided by different cloud management software

Type	Cloud management software	Cloud interface
Commercial	Amazon EC2	Amazon EC2 API, Proprietary API
	Microsoft Windows Azure	Amazon EC2 API, Proprietary API
Open-source	Eucalyptus	Amazon EC2 API, Proprietary API, OCCI API
	Openstack	Amazon EC2 API, Proprietary API, OCCI API

The OCCI API contributes a general API that can be integrated with different cloud management software. It also provides the functionality for performing both the virtual machine management operations (virtual machine management) and the virtual network management operations.

2.10. Related works

In recent years there has been a number of researches done in the area of cloud computing. But few researches which test and measure performance of cloud computing service was conducted. Several papers also covered some concepts about testing as a service. The gaps of the thesis; methodology used, their initiation; results gotten in the previous work was revealed in the next paragraphs. At last the contribution of the study is added.

In Niloofar Khanghahi and Reza Ravanmehr [27] an overall perspective on cloud evaluation criteria and highlight with help of simulation was provided. The researcher's motivation is higher performance of services and anything related to cloud have influence on users and service providers. Due to increase in development of cloud computing, it is predicted that storage and computing on personal computers has been forgotten and all of these things will be transferred into the Clouds. Major factors of cloud computing performance are presented and analyzed. The authors use Cloud Analyst tool which is a graphical design based on Clouds for simulation determination.

The paper took different sample of users and requests per unit time. The paper concluded that if data center is overrated capacity by user, it will not only be unprofitable but it also reduces efficiency of the center. The same number of user requests is connected to one data center. The result of study indicates increment in number of requests per unit time has little impact on response time, on processing time and data centers. It is effective on the cost of data transfer. Increasing the number of request increases cost of data transfer. Increasing power and speed of the data center is not always efficient. Sometimes it only has additional costs. For this case, distribution of data centers and using closest data center is better and more optimal. The factors that are not covered in this study were considered in the researcher work. Additionally, testing performance of cloud computing continues activity because of development of cloud computing and users request increased. Other measures will be conducted in this study since cloud computing still has shortages in performance; for instance, selecting infrastructure configuration type and cost calculation system.

In Sinung Suakanto [28], the authors discussed that cloud computing relies on the use of an elastic virtual machine and the use of network for data exchange. The aim of testing the quality of cloud

service is the fact that most providers push number of users to use their services but don't pay attention at the declining quality received by the average user. In this work the researchers conducted experimental setup regarding service quality gained by the cloud users. To analyze the performance of the cloud, the criteria received was number of users and number of requests per period of each user. The number of each user access single application service and all traffic directed to the same VM of cloud computing. The interval time used between requests is normal distribution. This means number of requests per period for each user is assumed equal and the difference is the interval between requests is in a random process. The proposed tool used was eucalyptus as an open source cloud computing and they used HTTP protocol in case of evaluating how the quality service was provided by Eucalyptus.

The hypothesis indicates that increasing number of users will decrease the average performance received by users, which is a problem on quality of service (Qos). The authors propose to use specific model of cloud computing architecture with specific service running on it. As usual, the data was subscribed on external cloud, and the users of the application were from the same enterprise. So, with this model, most of the users of the service application came from the same network or region and also members of the organization. Users' request repeatedly by the process called stochastic approach. To examine and have comprehensive result, four setup experiments had been planned in different network regions. The main difference between the authors and our thesis is that the authors focuses on performance testing at the eucalyptus level whereas our thesis focuses on the selective testing for CC as a general by deploying local cloud using openstack platform and Application Manager tool used to test Amazon via online testing.

In Jerry Gao, K. Manjula [29] proposed testing as-service (TaaS) infrastructure and reports a cloud-based TaaS environment. TaaS concepts and CTaaS have been presented, including their infrastructure, design and implementation. In addition, the paper demonstrates the application for previously proposed graphic models and metrics for SaaS performance and scalability evaluation. The architecture has been designed to describe and show selective testing insight and they tested virtual machines of Amazon EC2. Jmeter and selenium used to analyze performance and conduct simulation parameters such as the number of users, traffic loads, starting and ending time, and delay time intervals between loads. The other contribution of the paper was a finding that different performance testing metrics have been measured. Students perform test suites based on pre-defined

test models with detailed metrics and parameters. The person who tests controls continuous execution of the same test suite for a selected testing time period. The paper lacks approaches for general selective testing system. Testing as services is continuous process in cloud computing.

In Mumtaz M.Ali AL-Mukhtar [30] evaluates Eucalyptus and CloudStack performance in terms of CPU utilization, memory bandwidth, disk I/O access speed, and network performance using suitable benchmarks. The authors evaluated VM management operations such as add and delete were assessed to determine which cloud deliver high performance. Simple web application was implemented to evaluate their suitability in web application hosting. The research was motivated by the fact that the companies did not have any knowledge of where their data and applications run and their data stored.

Nowadays, when service failures occur, public cloud providers only offer to refund their customers regarding the infrastructure outages; they are not inclined to pay for outages service and low performance level that would refund customers for loss of business revenue. Providers are not only required to supply correct services but it is to meet their expectations in the context of performance. They put advantage of evaluating cloud computing performance like exploring the limitations of the cloud platforms and improving Qos, infrastructure planning, and making a wiser selection of the platforms. Our study has to be evaluating performance of cloud computing using some methodology, and benchmarks, other performance metrics like scalability which are not tested by previous researchers. To expand our study, the researcher experienced performance of local cloud in addition to general selective testing of cloud computing.

In Vijayasanthi M1 [31], the authors tried to identify factors that affect performance of cloud computing. To study the performance of cloud and find solution, they used Eucalyptus by using KVM hypervisor. Different tools were used to demonstrate performance of eucalyptus. Every experiment was performed with two running virtual machine instances. To analyze capability of Eucalyptus, they took 15 GB disk, 512 MB RAM, and 1 CPU core using iperf simulation tool. A simple program was created to check processor sharing between instances. This program consists of a while loop which calls function and saves the output in a text file. Flows of UDP packets, packet loss, RTT variations and jitter results were obtained. The test involves incoming and out coming packets and conduct each part of the experiment on both Vms. The finding of paper was limited on the Eucalyptus but in our work we bring selective testing interface. This is very important since any

user who needs to know performance of cloud computing, about cloud service, where cloud data center exist, what kind of operating system supported and cost for each instance type have been discussed and addressed very well.

In Robayet Nasim [32], the authors revealed as a problem, there is no clear indication about the performance impacts of underlying infrastructure. They have deployed OpenStack private cloud over both non-virtualized and virtualized environments and evaluated its performance in terms of CPU, bandwidth, and data transfer rate. In order to perform the study of physical setup experiment; they use hardware and virtualization comparison; iperf tool to measure and test openstack and finally results analyzed using OpenStack which is deployed over dedicated hardware always performs better than OpenStack running over virtualized environment. The previous work uses openstack CMSP. The researcher study the performance of cloud based on local cloud version kilo and deploy different nodes with different number of interfaces on each node. This can be tested in terms of scalability, latency or packet delay. In addition to previous work, the researcher measured performance metrics.

In Swapna Addamani, Anirban Basu [33], the authors suggested that different approaches to performance analysis of cloud computing platforms and proposes model. They proposed queuing model architecture. The model shows, the cloud computing platform is modeled as multiple queues and the VMs are designed as service centers. This means to test performance, VMs stores application which runs on servers. The analyzed result could be captured using JMT tool. The author used Virtualization software such as Xen, KVM to run an application through cloud and to observe performance. The virtual machines of an application run on either one cloud node or multiple cloud nodes, and each of them has the same computing resource to see the difference, comparison and evaluation of service. The author release tourism management system and product search engine which were run to measure parameters. Problem of the research was users submit their requests for computing resources like CPU, RAM, disk, application, infrastructure and software which are provisioned in the cloud without the users being aware about the details of execution environment. The researcher saw the shortage of paper and brings proper awareness for users by the way of selective testing of cloud performance. This could be done through analyzing of computing resource request of number of users by given of constant access time and create single CID.

CHAPTER THREE

3. Proposed System Design

To attain the objectives of this research, different methods and techniques was used as already discussed in chapter one. The next section presents methods and techniques of the current state of cloud computing systems measurement, the kinds of measurement tests that can be performed on the systems and the procedures used for structuring the test model. The researcher designed a performance evaluation method that allows an assessment of clouds. To this end, the evaluation procedure was signified. The steps show selecting specific cloud providers for case study and the computing resource (i.e. size of VMs) used. Cloud computing service made to run on the virtual machine, not in physical machine as usual. It may lead to create much identical application service that could manage more efficiently. The type of operating system used to run the experiment, type of cloud evaluated and the tools important for this study to capture data when the application is running on the cloud are described in detail in the first chapter. The more details of methodology and proposed system work are discussed in this chapter.

3.1. Performance evaluation steps

Our performance evaluation model is based on tenant virtual machines system benchmarking for individual VM as well as the whole system of cloud. It comprises five major steps as shown in figure 3.1. The system starts by selecting infrastructure as a service for sample to test model of the cloud and build the cloud environment from each CCP for the same set of hardware resources on the laboratory environment.

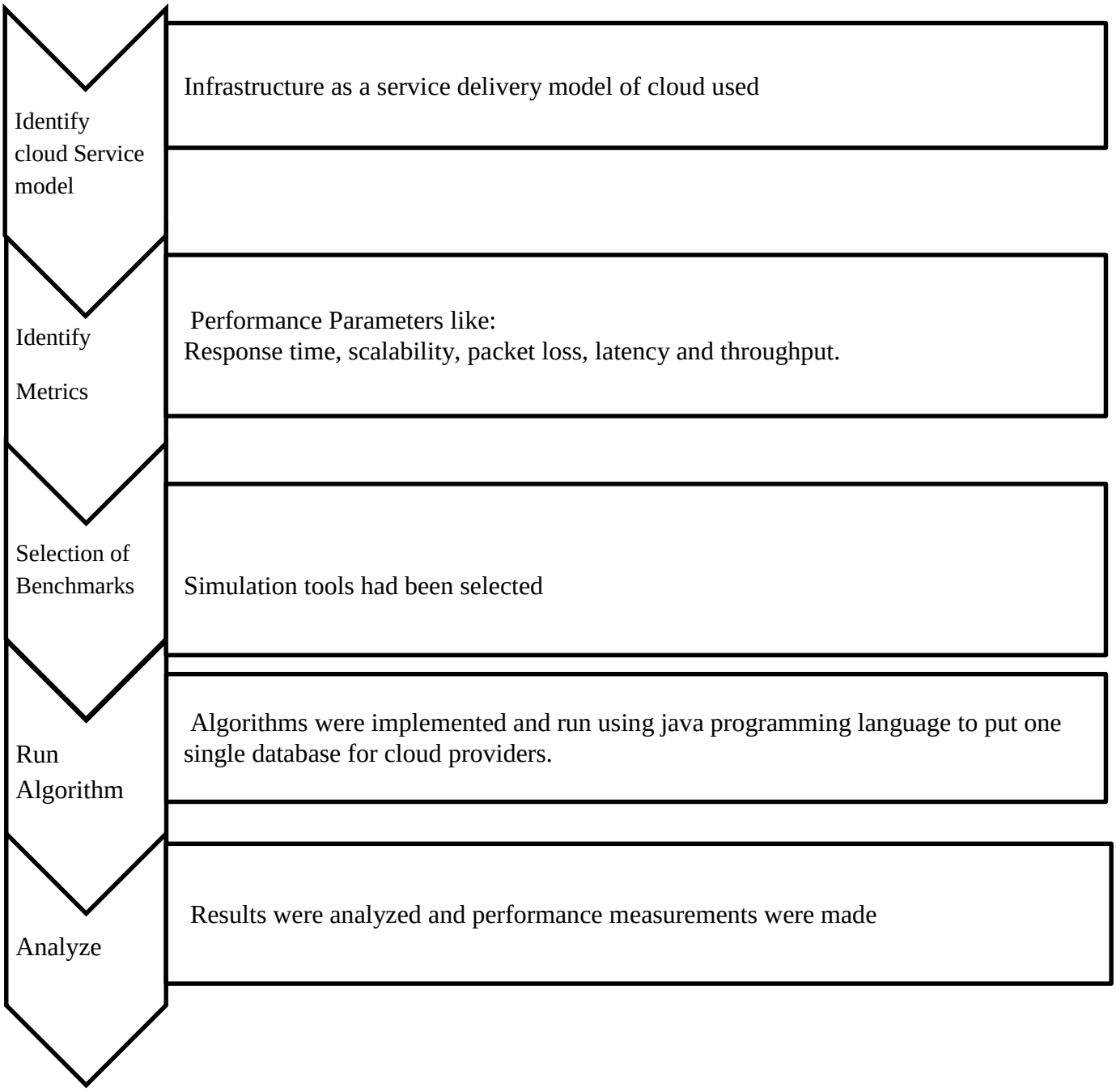

Figure 3-1 Model of evaluation Steps

Designing interactive system

For the designing of graphic user interfaces and test the application of users the language used is java (JSP) programming language with MySQL database at the backend. By using these tools, an interface called STFCC has been developed. More detail that the user gets through the information has been displayed by the designed interface. This form of evaluation methodology helps users to confidentially subscribe their application in the cloud. The risk and frustration of users minimized when the technique of selective testing is applied.

System components

The function of cloud is providing services for users of the world who are connected with it. When we need to study selective testing in cloud, considering components of cloud computing is not forgettable. Cloud user, interface, cloud database and cloud computing are main focuses throughout the work.

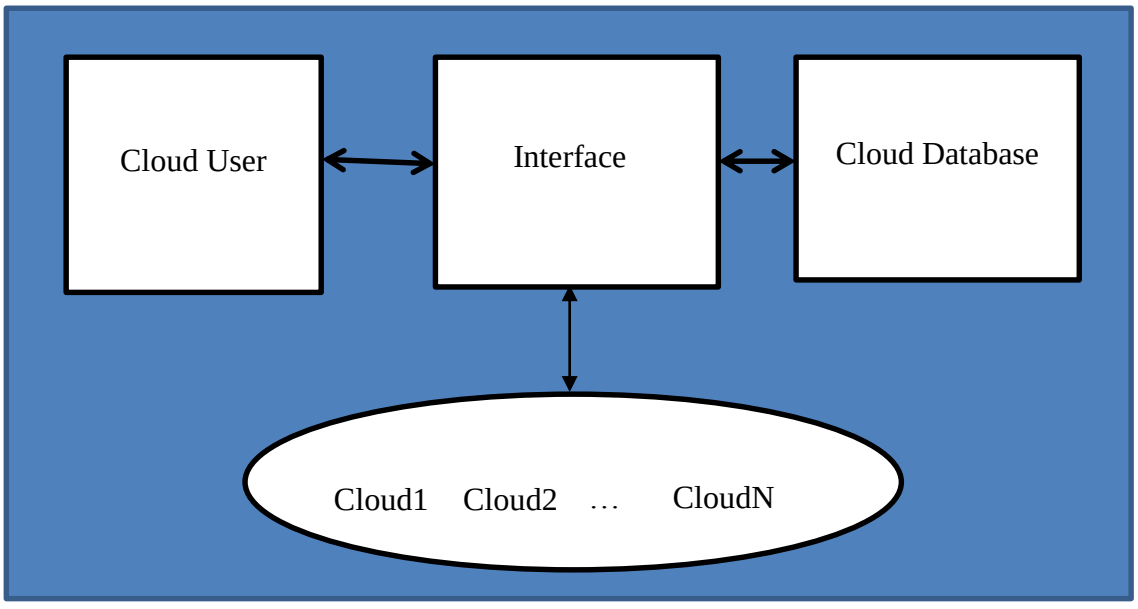

Figure 3-2 Components of Cloud Computing

Cloud tester who indirectly interacts with the system gets information about the available flavors on the system and the instances running on them. Cloud tester also specifies the configuration of the program. The user contacts the system cloud administrator to get more information about cloud computing.

3.2. Proposed Algorithms

The researcher prepares testing as service for cloud computing by means of different techniques and methodologies. The procedures applied are described by algorithms used.

Interactive interface

This algorithm uses to design a user interface for interacting with cloud computing. Users are not familiar with cloud computing. So, it is important to make the interaction with cloud as easy as possible. It is very vital for the interest of the users as well as to have confidence to upload their application in the cloud. So, STFCC acts in the middle of the user and cloud system and reach cloud information to customers. The proposed algorithm brings all performance of cloud providers in a single database.

Performance of selective testing for cloud computing pseudo code:

Algorithm 1: Interactive interface

Input: vCPU, RAM, HardDisk; time; number of users noOfUsers; cloud providers cp; cloud providers performance cPp;

Output: Performance P (throughput, response time, latency, description of cloud providers).

1. begin
2. For all cloud providers cp getcpPInfo.
   loop1: Choose cp
3. cloudcp1 ⟵ Test
   loop2: if (VM= vCPU, RAM, Disk) && (tested based on time, noOfUsers, application)
   For user has App1, App2…Appn
   VM ⟵ put (Application)
   NumberOfUsers ⟵ send request
4. cp1 (Application) ⟵ run
5. P ⟵ respond then
6. if (user satisfied to providers performance)
7. Choose cp1
8. if (user not satisfied to cp1 performance)
9. goto loop1
10. cloudcp2 ⟵ Test
11. goto loop2
12. cp2 (Application) ⟵ run
13. P ⟵ respond
14. If (cp1P> cp2P)
15. Choose cp2
16. Else
17. goto loop1

User requirements

STFCC deals between user and cloud to give testing as a service. Users' requirements collected from users those who need to know cloud performance and subscribe their application. The users always require fast access in order to appropriately run business organizations. Cloud providers are business oriented, though to know which cloud provider give fast and best performance the users can use STFCC system. The system collects detail business requirement of users including amount of CPU, bandwidth in different speeds, memory RAM and storage of hard disk. Then, fill type of users' requirements in each cloud provider and display performance of each and every cloud providers' service for users. Decision of users depends upon cloud performance they have seen through interactive service.

The following statement shows pseudo code for users' requirements:

Algorithm 2: User requirements

Input: computing resource, no. of users, time, application

Output: SaaS, PaaS, IaaS provider performance

1. begin
2. For users need=Service type
3. If (users==IaaS)
4. display_Providers ← IaaS (list) then
5. Provider ← choose
6. requirement_list(computingresource,no.ofusers,time,application) ← executed_list
7. performance ← read then
8. Else if (users==SaaS)
9. display_Providers ← SaaS (list) then
10. Provider ← choose
11. requirement_list (computing resource, no. of users, time, application) ← executed_list
12. performance ← read
13. Else

14. (users==PaaS)
15. Providers ⟵ PaaS (list) then
16. requirement_list (computing resource, no. of users, time, application) ⟵ executed_list
17. performance ⟵ read
18. End if

Calculate cost and respond providers pricing policies

Cost saving is one of the main purpose that enterprises or users consider when migrating their application to the cloud. For this reason, cloud consumers able to make accurate estimation of cost through selective testing process. A question that comes from user is pay-per-use charging model. So the question is how much will cost me to run one application service in the cloud for a given day (s), knowing that it may needs 90 GB of storage hard disk? When users bring their requirements such as memory capacity, bandwidth, and CPU speed and disk storage capacity, the system shows performance. Next to this, other link is displayed. It demonstrates total cost depending on the cloud vendor management and cost pricing system. Normally, cost covers usage duration, volume, read and write operation for storage on the disk, size of data incoming or outgoing. Mechanism of cloud providers' calculation depends on type of instance they give. Size of users' application decides instance type they are going to purchase. There are large, medium and small types of instances. These kinds of instances are associated with cost. When virtual machine was created for users to host their application, cloud manager decides instance type as a result of information gained from subscribers. Different providers have their own pricing policy. The way of calculating cost algorithm is designed according to vendor's policy. For example some cloud providers calculate cost based on type of virtual machine rented while other vendor is based on the pay-per –use, or access per hour, per month and per year.

The next algorithm is applied for cost calculation based on computing resource and providers' price policy.

Algorithm 3: Calculate cost and respond providers pricing policies

Input: cloud providers name cp; virtual central processing unit (cpu) vCPU; RAM; HardDisk; API; region;

Output: Cost of cloud providers

1. begin
2. For cost ⟵ price of cloud providers
3. if (price calculation policy based on provider)
4. Cost calculation = select policy
5. Configuration ⟵ type(m1.small, t1.tiny, m1.medium, large)
6. VMid ⟵ given ID
7. Size ⟵ resource(vCPU, RAM, HardDisk)
8. cloud ⟵ API
9. Region ⟵ datacenter(location)
10. cost ⟵ rate
11. if (cost=perhour)
12. totalcost=resource+contract period
13. cost ⟵ rate
14. Else
15. if (cost=perday)
16. cost ⟵ rate
17. goto 12
18. Else
19. goto step 3-to-step 10
20. End if
21. End for

With regard to solving user's problems, we intended to dig out cloud providers' pricing policies. The steps stated show that certain users may tend to pay per hour access while others may prefer to pay per year. Because of such variance of users' interest, system designed shows every provider's pricing facet according to users request and requirements.

3.3. Approaches of selective testingof cloud capacity to evaluate performance delivery

The diagram below indicates selective testingof cloud capacity to evaluate performance delivered in given time. In the case of evaluating and testing technique, the researcher designs the model of measurement methods by involving different components. As the figures 3-4 try to illustrate, the following elements are included.

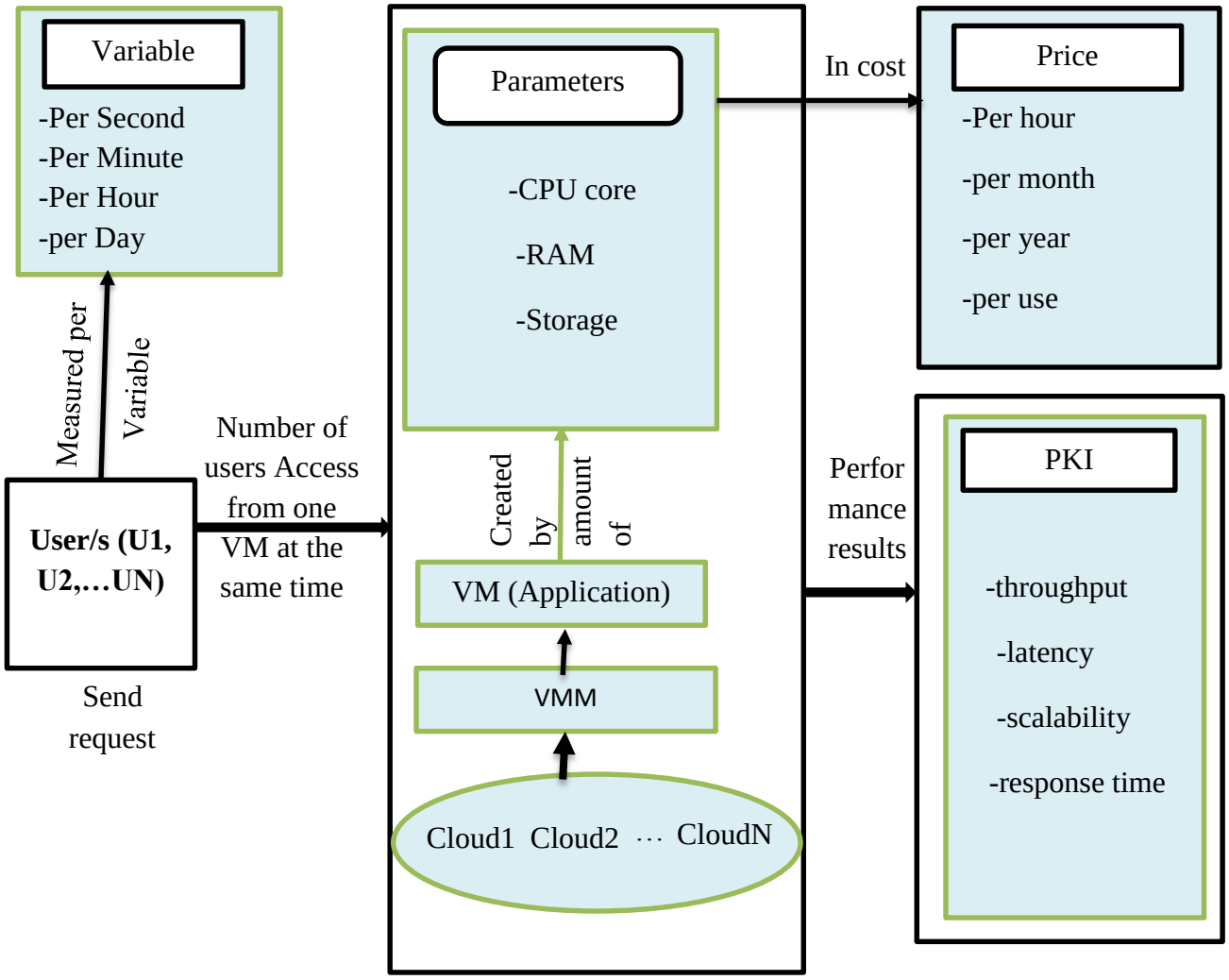

Figure 3-3 Algorithm of selective testing of cloud capacity

Variables: At the time of accessing the cloud, certain number of users send request to the virtual machines. We can gather performance of cloud by the time. For instance constant number of users sends requests to the cloud (let say 1000 users). These user starts mining data (may be per second, per minute, per hour, and per day).

Users: These are consumers of cloud providers' service. The researchers tried to designate the users' requests within fixed time or simultaneous time in the diagram. This shows that we can shortly identify performance of cloud provider's service deliver to their customers. Number of users is changed with constant time to see cloud performance. After we have got the detail response time,

throughput, scalability and latency of cloud, we can decide which cloud provider deliver best performance than others.

Parameters: This is the basic element for the presence of infrastructure and it is one part of service delivery model type in CC. There are number of virtual machines created in the CC. These virtual machines are designed using different characteristics. These characteristics are called parameters; basic building blocks of VMs. CPU core, RAM, hard disk and bandwidth are basic things to run machines and hold users data. So, virtual machine is developed from constant number parameters (for example: 3.3 CPU MHz core, 2RAM in GB, 250GB of hard disk 100mbps of bandwidth).

Performance key indicators (KPI): The main aim of this research is to identify which cloud providers give best performance for services consumers of cloud computing. Hence, as it has been proposed, system of selective testing is developed. All cloud providers do not give equal performance since different factors can affect network transmission. But users' are interested to get high quality of service. Qos is evaluated using throughput, response time, scalability, latency and else of criteria. Testing techniques solve challenges and risks of cloud providers' services. Depending on the cloud providers' performance, users can assess what is key performance indicators value *via* cloud type to subscribe application as it has been needed.

Price: There is a mutual benefit between consumers and providers. When users put data in the cloud, they get advantages like storage, backup, cost reduction and else. Besides, providers collect cost from users based on the criteria of providers' pricing techniques. The infrastructure (server, NIC, RAM, hard disk, data transfer) needs money to buy. Since data of users use these devices, the owners of infrastructures must get money. In this study, the researchers need to show costs of providers, because pricing style of all providers are not unique. Users need minimized cost. For this reason, pricing policies of providers had been discussed by collecting from different sources to value users.

The above figure changed to pseudo code.

Algorithm 4: Predict cloud performance

Inputs: userReqList: Set of user requirements,
Time: time needed by user to access cloud
NoOfUserList: Set of constant user access cloud
cloudList: Set of cloud descriptions,
instanceList: Set of instances,
cpuCoreList: Set of CpuCore,
RAMList: Set of RAM
HardDiskList: Set of HardDisk,
userReq: Resource
NoOfUser: Amount of user
Cloud; VM: Resource

Output: totalCost: Estimated total cost for an instance
Performance key indicators of cloud

Start

For (every cloud in the cloudList)

InsatanceList ⟵ get

For (every Instance in the Instance_List)

vm ⟵ user (request) then

Calculate ()

cpuCoreCost for (vCPU);

RamCost for (RAM);

HardDiskCost for (hardDisk);

totalCost = (cpuCoreCost+ RamCost + HardDiskCost)

return totalCost;

End For

Function PKI ()

for(every cloud in the cloudList)

cloudList ⟵get

vm ⟵ user(send) then

measure ()

throughput for (cloud, time, NoOfUser);

responseTime for (cloud, time, NoOfUser);

latency for (cloud, time, NoOfUser);

output = (throughput, responseTime, latency)

return output

3.4. Cloud service identification

The proposed and developed system has details of cloud services. Users look for different type of cloud services according to their interest and business. When selective testing for performance checking is developed, the cloud type designed with which type of service they give. This means software as a service (SaaS), platform as a service (PaaS) and infrastructure as a service (IaaS). Before subscription to a service in cloud, performance of the cloud should be identified. The system can perform this operation because the system has different pre-user requirements. The person who intends to subscribe his application checks the cloud performance to identify the better service.

What the user informed about the cloud providers system?

- Cloud providers name
- Location of cloud providers or data center
- Supported operating system and programming language
- Pre-users requirements
- Cloud providers service type
- Rate and estimated cost of cloud providers
- Performance of cloud providers

Selective testing monitoring design view

Cloud monitoring techniques used in this research is very important especially for users because selective testing design provides aspect of cloud and avails with usability for users. When testing is processed at the presence of users, the following designed system reports feature of cloud depending on the requirement types collected from customers.

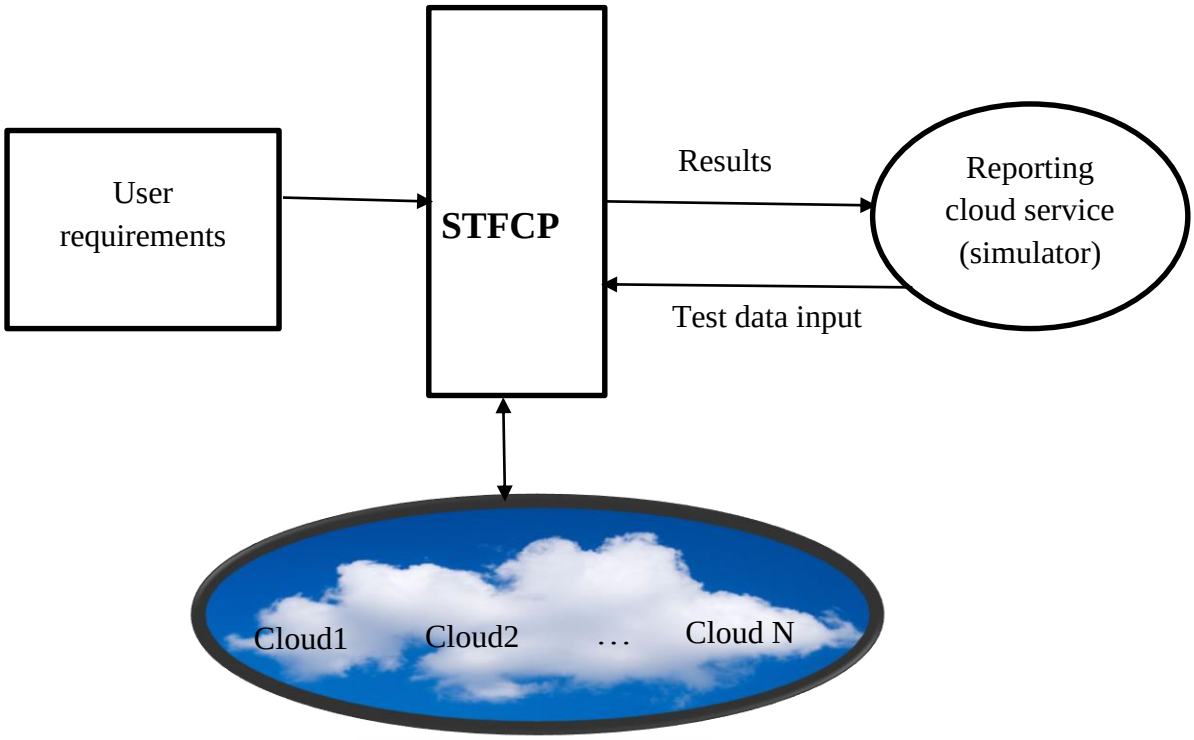

Figure 3-4 Design view of testing as a service

In the above diagram, requirements of users are processed to find cloud performance. This could be handled by selective testing and testing approaches by designing STFCC architecture. In the above figure, users interface is developed which accepts different requirements information and test cloud providers' service delivery. In this case, all requirements are filled and tested based on cloud vendor capacity, cloud data center, service type and pricing policies of cost calculation. After performance of cloud is tested based on user requirements, the report will be displayed.

3.5. Cloud Information Database (CID)

The researchers find about cloud providers' performance and data to store in one database. Cloud services collected from different parts of service providers that to keep and represent in a database. Cloud providers have their own rule and system of service delivery techniques. They may differentiate by service type, type of hypervisor they use, programming language their software support and written with, location of data centers and performance they give. To compare all performances of services given by clouds, it is essential to have database by integrating with cloud, and front-end tool.

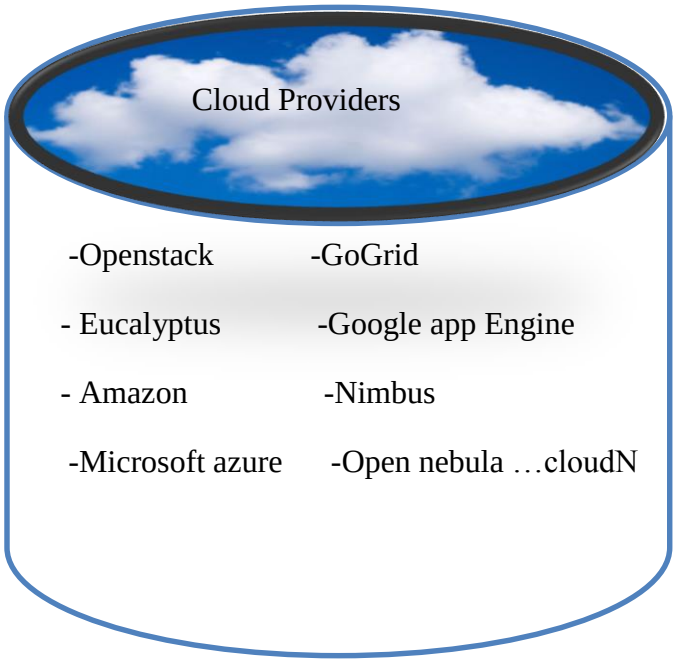

Figure 3-5 Cloud Providers Database

Since large of cloud providers are available at this time and it is difficult to gain comprehensive information about them, we bound to some familiar cloud provider types which are to be included in the database. Another reason is related to impossibility of installing all cloud provider software and testing their performance since there is shortage of machine; time and some part of them are commercial. As discussed in this chapter Local cloud is tested practically in the laboratory environment and has got its performance quality and status in more detail.

CID contains numerous tables related to the service of cloud computing. As deliberated in the review of current practices, different providers have dissimilar policies which differ from others by far. In the designed system, however, all providers' information should be reliably stored in a single database to display performance, to calculate the expected cost without help from extra programs. To achieve this, each module containing of the high quality performance is primarily defined as a single table, including provider, Parameter Sets, Operating System, Datacenter and Service Type table. As parameter set is dependent on the provider, it has a foreign key referring to provider table.

Then, PKI table is demarcated with foreign keys to all components but actual pricing elements are stored in a separate table from original performance key indicators that were measured and evaluated, named PKI, PKIList table. The PKIList table has three fields: parameters, price and period. Indicators are throughput, response time and latency. Price is the details that has been calculated depending on the providers pricing policy system and added to the total cost on every given period repeatedly for a whole usage time. At the end database has to be refreshed when there is change on a provider's service, if there is new comer of cloud providers to database and if a provider stops giving service.

Algorithm 5: refreshing database

Input: cloud provider cp; API; VMid; vCPU; RAM; HardDisk; instance type;

Output: Database

Refresh database

1. Begin
2. for new cloud provider
3. if (cloud is new cloud provider)
4. for API, VMid, vCPU, RAM, HardDisk, instance type
5. cp ⟵ VMM(register)
6. API(VMid) ⟵ API
7. VM ⟵ VMM (run)
8. Database ⟵ update
9. for (cloud provider in CID)
10. if (cp_change = service) then
11. Data ⟵ load
12. Rate=”perday”
13. Database ⟵ update
14. Else (cp = changed)
15. cp ⟵ delete (from database)
16. End for
17. End if

3.6. Design architecture of selective testing for cloud performance testing

The proposed model retains the cloud application that assists on performance evaluation. The proposed cloud computing model is comprised of a front end and back end architecture. These two elements are connected through a network. The front end is the vehicle by which the user interacts with the system and the back end represents cloud database which contains cloud information. The

front end is composed of a client computer, or computer network of an enterprise, and the applications used to access cloud. Back end will provide the required services in a required manner.

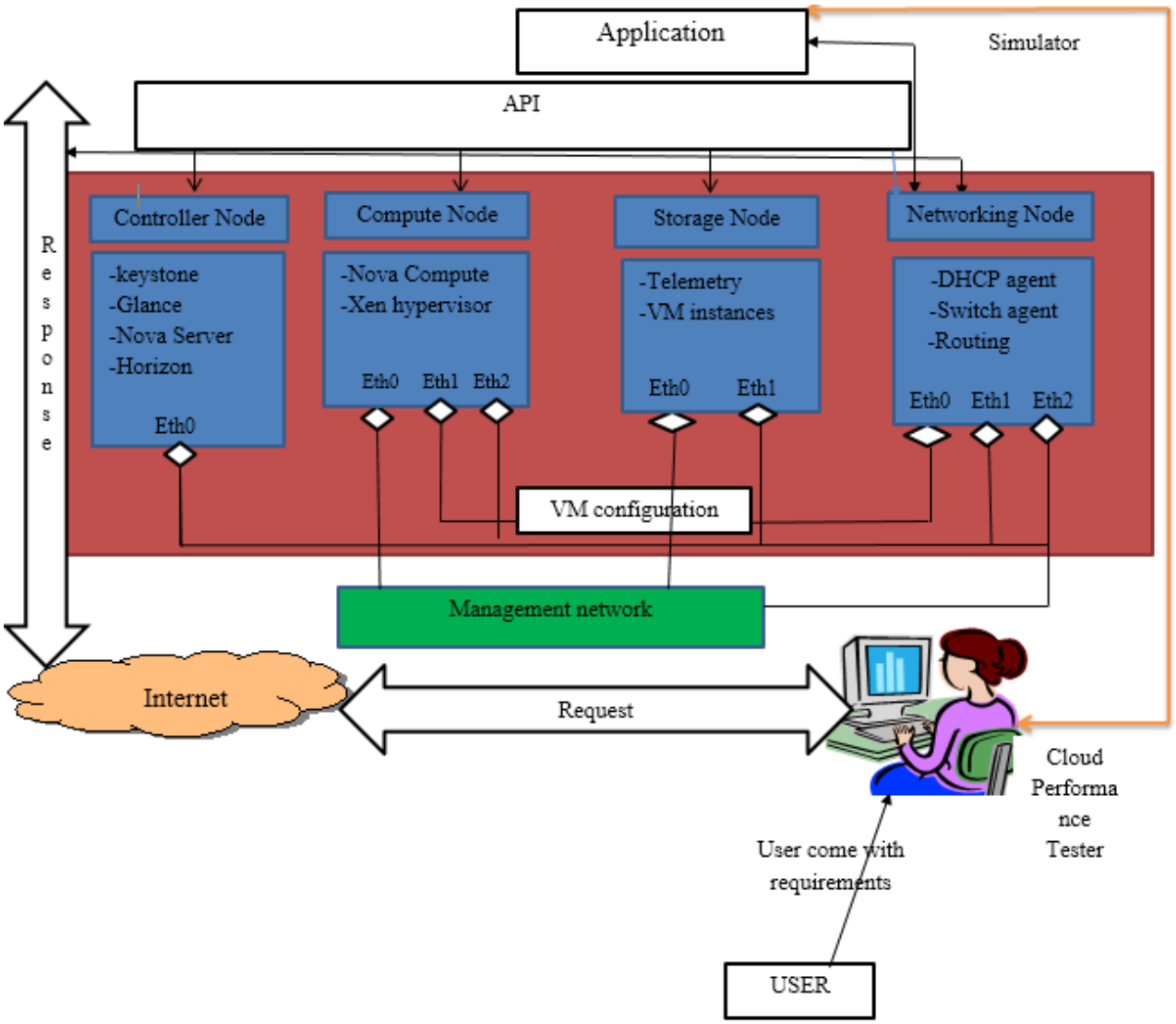

Figure 3-6 Proposed cloud performance testing architecture

A diagram with the whole system which is drawn to develop thesis structure is shown in the above figure. Once the VM image has been transparently deployed on a physical machine, the resource status is running as the instance is booted. At the end of the boot process, the resource status will be installed. The installed resource can be used as a regular computing node immediately after the booting process is completed through connection.

The figure 3-7 refers skeleton of the thesis procedure, where request and response communication is performed. With this scenario, simulation tool has to run to look at the behavior of the packets going through the network. Afterwards, how the network performance was different when some factors could be changed (number of clients, type of instance, etc.) is to be checked. The traffic was recorded in the network tool for a further analysis and pattern evaluations. Finally, with the pattern obtained, the researcher comes to the last part of the traffic pattern gathering of key performance indicators. In this part, many data sources and many requests are set up to repeatedly recreate the traffic pattern towards the cloud. By this way, cloud performance is found when it is able to handle heavy traffic loads and requests come inside from different data sources as well as responding repetitive request of data.

Application

Large data size can be stored in the cloud computing. Through the process of virtualization techniques, hardware of physical machine can be shared into several virtual machines. On the virtual machine, data has to be hosted and accessed by user. In this architecture, application sits on VM. When user access that application, cloud tester look at throughput, response time, latency and cost.

Application Programming Interface (API)

Interface through which administrator and user get and access cloud is known as API.

Network that capture KPIs

While the application is running and accessed by different group of users, cloud tester gather and write down what performance has been given by cloud through tool.

VM configuration

Type of instance created in the cloud providers has different names. For example Amazon EC2 VM names are: small, large and extra-large [34].

Cloud performance tester

To analyze network performance and packet flow situation, the cloud tester has responsibility to capture and see providers' strength and weakness. Identifying and analyzing different cloud performance providers gives confidence for users in hosting application.

Internet

Cloud computing is the delivery of computing services over the internet and it is metaphor of internet. Cloud computing is nothing without internet. The availability of internet determines existence of cloud. In the diagram above, we understand that internet link a user and administrator with cloud computing. The cloud computing model allows access to information and computer resources from anywhere that a network connection is available [35].

Request

Through http/s and URL of cloud providers, users send request to cloud. Generating more load to cloud shows about cloud providers' performance such as scalability. By stressing cloud servers, we can analyze cloud performance.

Response

It is known that if there is request, there is also response. Users access different cloud like Amazon, Google, Facebook, Microsoft azure, etc. The researchers put application on the virtual machines in the case local cloud and test response time, throughput and latency which also applied in other cloud providers.

Management network

Management network is a separate network for use by cloud operators. Naturally it consists of a separate switch and NICs, and is a recommended decision. The purpose of management network is for internal communication between cloud components. The IP addresses on this network should be reachable only within the component nodes to facilitate communication among them. This segregation prevents system administration and the monitoring of system access from being disrupted by traffic generated by guest computers.

Proposed interface design

Architecture designed for giving selective testing set solution for stress of users. The following diagram provides different services and presents cloud providers information. The system has the characteristics of accepting user requirement, display providers, service type they give, location of data centers and about cost policies. Different components designed in this interface have value and meanings for both customers and providers.

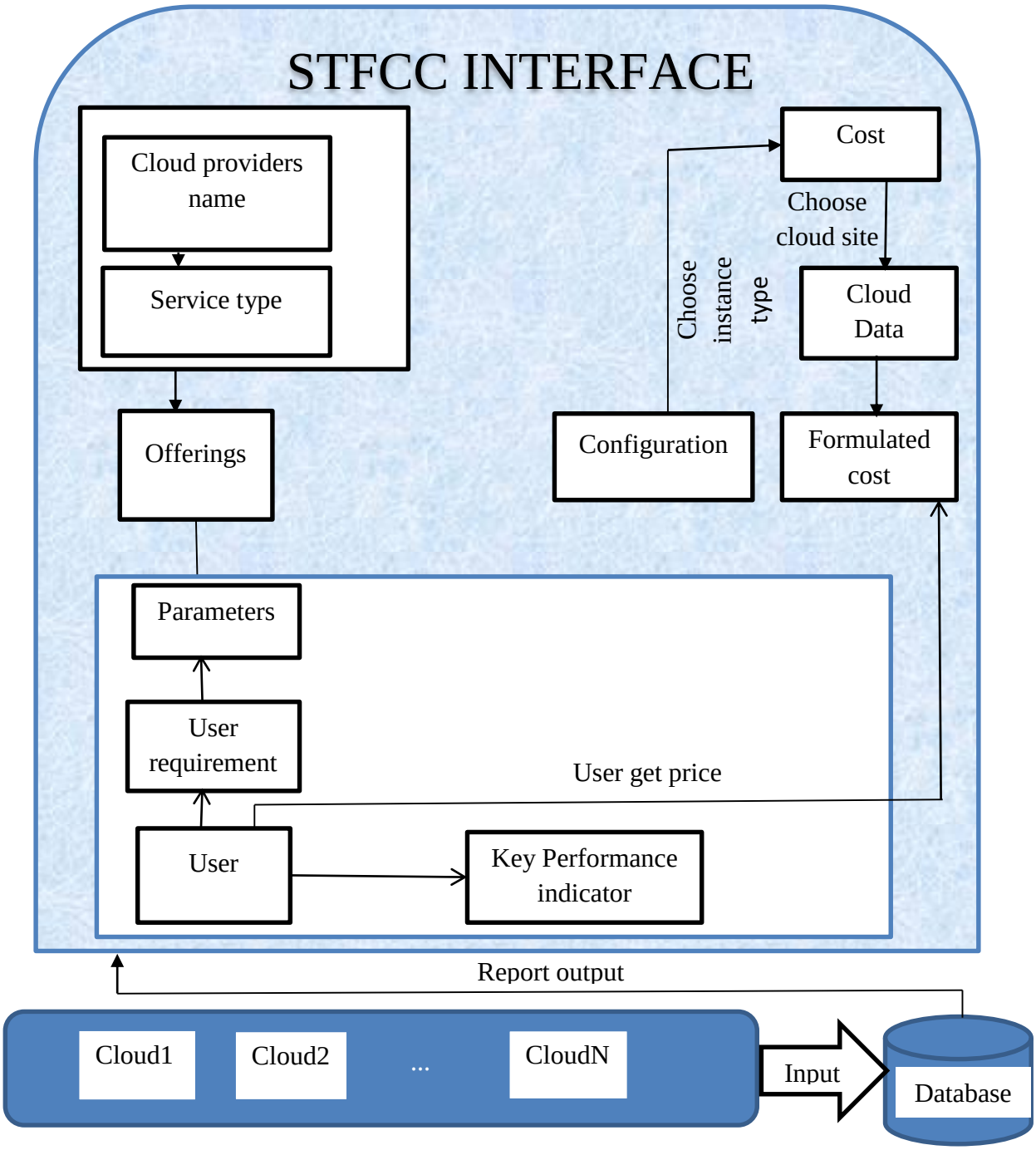

Figure 3-7 Interface design for decision system

Cloud providers type: - Today there are several types of cloud providers. For instance, Amazon EC2, Windows Microsoft Azure, Google App Engine, Openstack, etc. User chooses cloud providers based on performance delivery.

Cloud location: - one benefits of subscribing application in the cloud is recovering failure when one data center virtual machine failed. Price also reduces based on the location of data center for the user. If location of data center is nearest to user, the cost will be decreased.

Database: - All data and explanation of cloud providers stored in the database. Database houses different providers' aspect and features for later retrieval when user needs to join one cloud vendors for the purpose of comparing their performances.

Class diagram for selective testing

To bring best result of the study, algorithms are proposed. For the critical solution, algorithms are summarized by class diagram. Class diagram is preferable for testing as a service. Because class diagrams are useful in all forms of object-oriented programming (OO**)** and brings relationship and code dependency among classes. As tried to design in the figure below, there are different class names which were identified by their attributes and operations. Cloud providers were selected for the sample of the study. Among these local cloud and Amazon EC2 were well tested using different tools and the same computing resource to analyze performance difference.

VMM has the function of managing, monitoring, creating VMs, generating performance and set resource correctly. VMM interacts with cloud providers, database, and STFCC and Performance class diagram. Performance has been managed by VMM and assigned to VMs. The service of cloud tested through STFCC and display for the users by the help of interface. Database and VMM communicate and share information to each other. Every performance of cloud providers is managed by VMM and stored in the database. Tester pulls out result of cloud performance from the database. This means VMM fetch data based on the users action taken and then display PKI and cost for the users through interface.

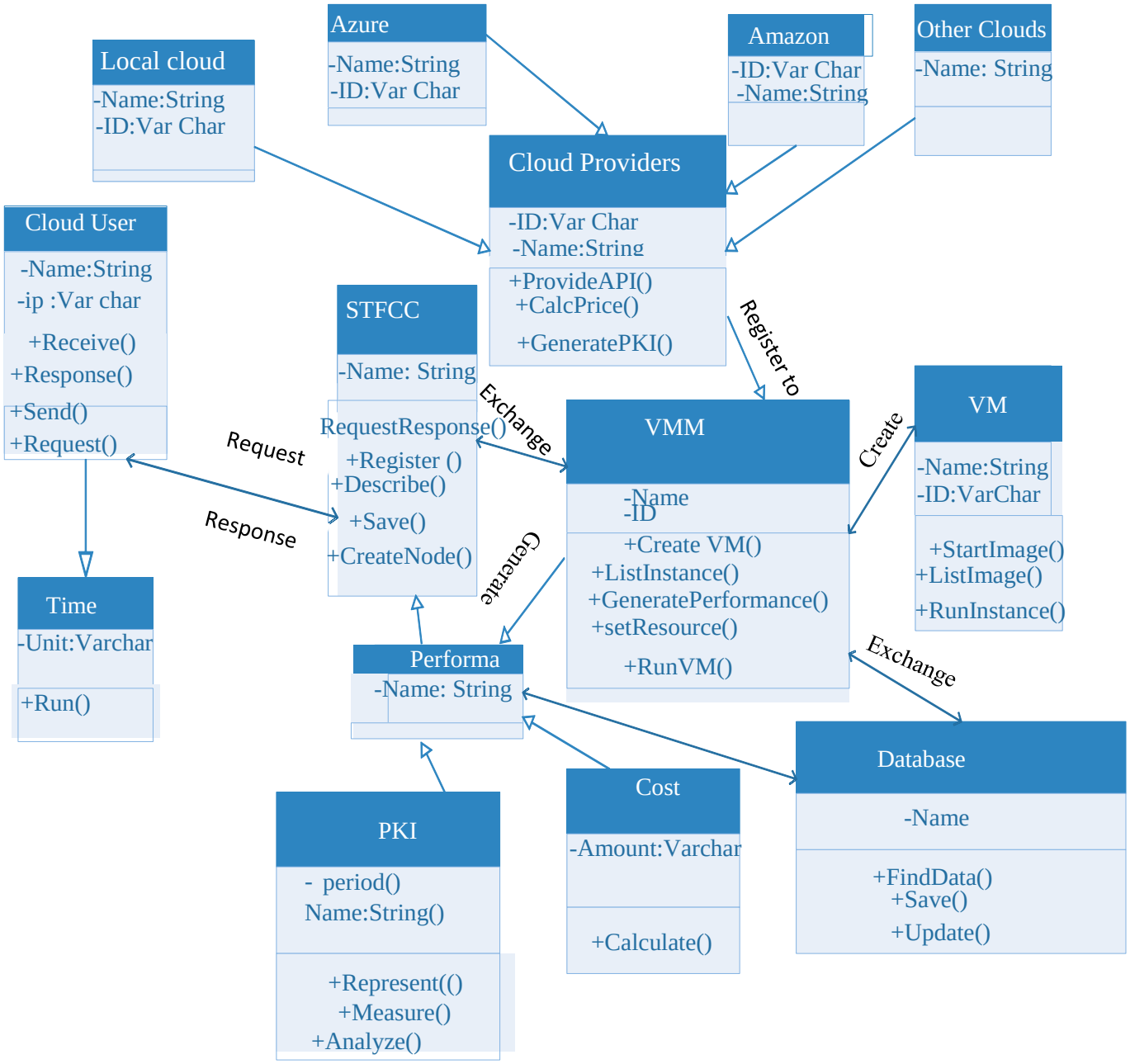

Figure 3-8 Class diagram of the proposed algorithm

CHAPTER FOUR

4. Experimentation

4.1. Laboratory test beds setup

Our study was started with identification of cloud computing that provide an infrastructure as a service. It is the most appropriate layer to deploy reference model and cloud framework. The experiment was prepared using open source cloud service provider. Openstack platform is one of them which have to be evaluated through the study [36].

The alternatives for setting up local cloud range from installing a development environment using the latest source version directly from local cloud repositories, usually from packaged versions of the components like the one that is provided by the Ubuntu package repositories. One must also decide on what components to install depending on the required functionality of the cloud being built. For this cloud, in order to serve multiple purposes as both lab test environment and an evaluation/demo platform, the choice was ultimately made to install from packages.

- To configure local cloud on laboratory environment, the researcher used different nodes which have unique tasks. These are:

Controller node: The controller act as cloud service manager that has the capability of managing cloud service applications. Identity service, image service as well as management portions of compute and networking, networking plug-in, and the dashboard were run on the controller node. Controller node was configured with one NIC, four RAM GB, 3.3 CPU, 1.7GHz and 500GB storage.

Network node: This could be run networking plug-in and several agents that deliver tenant networks and provide switching, routing, NAT, and DHCP services. Network node also handles external (Internet) connectivity for tenant virtual machine instances. The main task of Network node is tunneling purpose among the nodes. It deal with providing internet connection among nodes and pour the communication easily when internet connection is available. The machine has three interfaces and two of them connect internal devices and the third interface part connected to external

network. The node is configured with three NICs, two RAM GB, 3.3 CPU, 1.7 GHz and 500GB storage.

Compute node: compute nodes run different components. Hypervisor which operates tenant VMs or instances. Xen hypervisor used to configure, manage and monitor instances. Networking plug-in and an agent that connect tenant networks to instances. Telemetry agent to collect meters also run on this node. Compute node act as a user and on the machine supports to create virtual machines. The node is configured with three NICs, two RAM GB, 3.3 CPU, 1.7 GHz and 500GB storage.

Storage node: The Storage node contains disks that the Storage service provisions for tenant virtual machine instances. Telemetry agent runs on storage to collect meters. Network interface on storage node added to provide communication performance of storage services. The storage nodes contain the disks that the object storage service uses for storing accounts, containers, and objects. The function of storage node is storing data or application of users, because when cloud developed and deployed several tenants and volumes has been created. Virtual machine has the capability of storing data and application of users depending on the business requirements. The node is configured with two NICs, two RAM GB, 3.3 CPU, 1.7 GHz and 500GB storage.

Hardware requirements used in four nodes for local cloud configuration

The test bed was composed of four hosts. Table below shows in detail hardware requirements that were used in deployment laboratory test environment.

Table 4-1 Hardware Configuration of the system

Hosts	CPU	RAM	Storage	NIC	System Type
1.Controller	Intel(R) Core(TM) i3-3317U CPU@ 1.70Gh	4GB	500GB	1	64-Bit OS
2.NetworkNode	Intel(R) Core(TM) i3-3317U CPU@ 1.70Gh	2GB	500GB	3	64-Bit OS
3.Compute Node	Intel(R) Core(TM) i3-3317U CPU@ 1.70Gh	2GB	500GB	3	64-Bit OS
4.Storage Node	Intel(R) Core(TM) i3-3317U CPU@ 1.70Gh	2GB	500GB	2	64-Bit OS

4.1.1. Stating virtual machines in cloud computing

Virtual machines have been created for testing purpose on the local cloud. Cloud state is what the instance requests are being matched against to find optimal placements. The resource management system keeps track of the current cloud state and feeds it to the placement controller. Information about capacities of physical machines used to host VM loads and the network that connects them is used together with current VM allocations to find machines with extra capacity where new VMs could run.

Architecture of laboratory test

Before testing performance of cloud computing, we have to logically design architectural theory. This is very important decision to get perfect answer for our research questions which are stated so far. Nodes were configured with different packages installed on them which were connected with each other and with a device called switch which distributes and provides internet access. Mainly the figure below shows more detail.

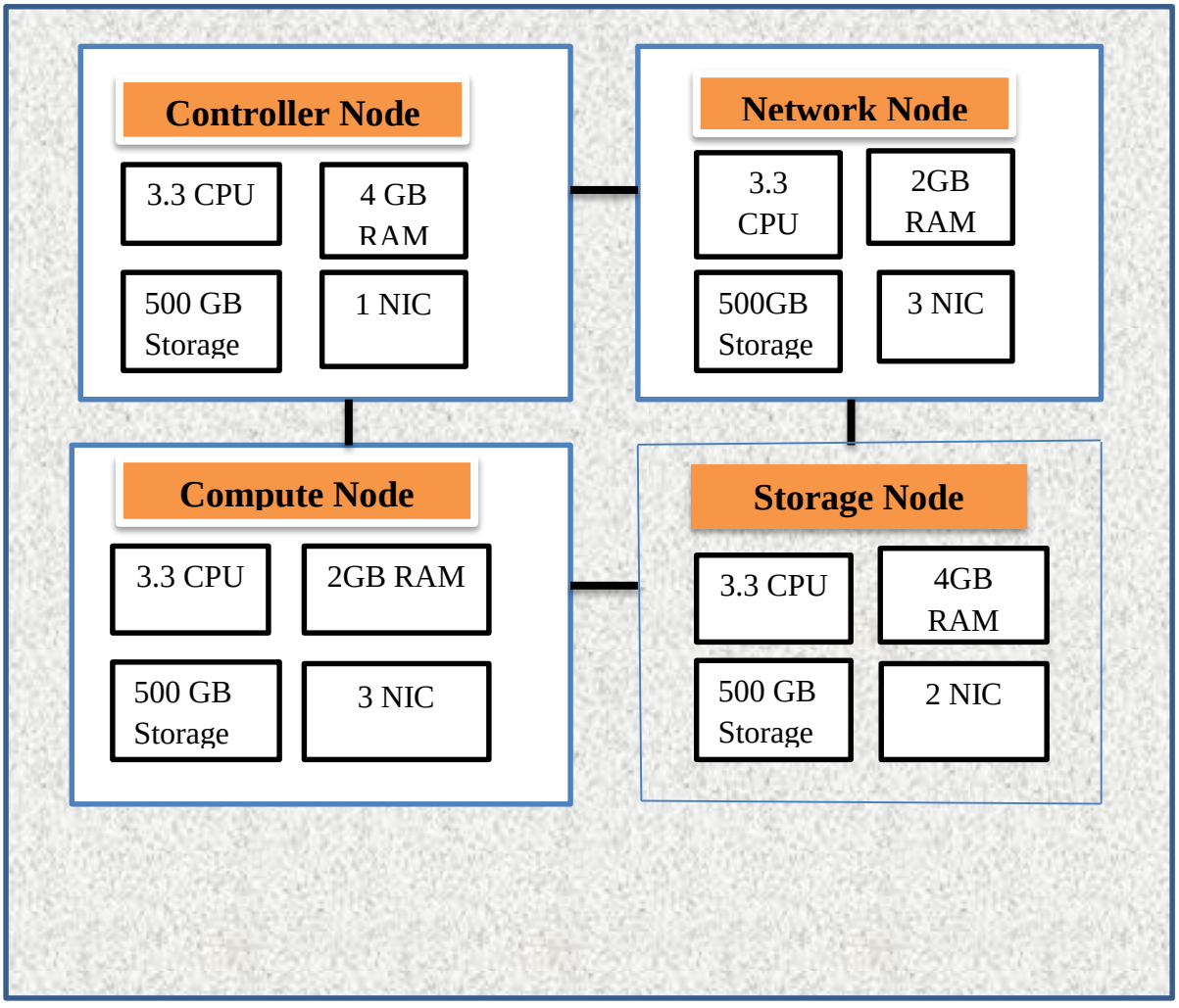

Figure 4-1 Design architecture of lab setup

Configuration steps on laboratory test environment

In above diagram lab installation, setup structured and different nodes with different tasks could be designed in laboratory test environment for the purpose of testing. Cloud prototype must be deployed to have deep test. Local cloud is highly configurable to meet different needs with controller nodes, various computes, networking, and storage options. Network interfaces should be configured.

The DEVSTACK Kilo shell script was used to install Local cloud on the target server node. Local cloud installation process is as follow:

1. Install Ubuntu 14.04 LTS version on the target server node
2. Install each packages on each different nodes
3. Clone the DEVSTACK installation software from Github source
4. Deploy the Local cloud by executing the DEVSTACK shell script on the laboratory environment.

4.2.Local cloud networking architecture layout.

The architecture layout with Local cloud networking (neutron) requires one controller node, one network node, and at least one compute node. The controller node contains one network interface on the management network. The network node contains one network interface on the management network, one on the instance tunnels network, and one on the external network. The compute node contains one network interface on the management network and one on the instance tunnels network.

- The next figure shows how each node should be configured securely.

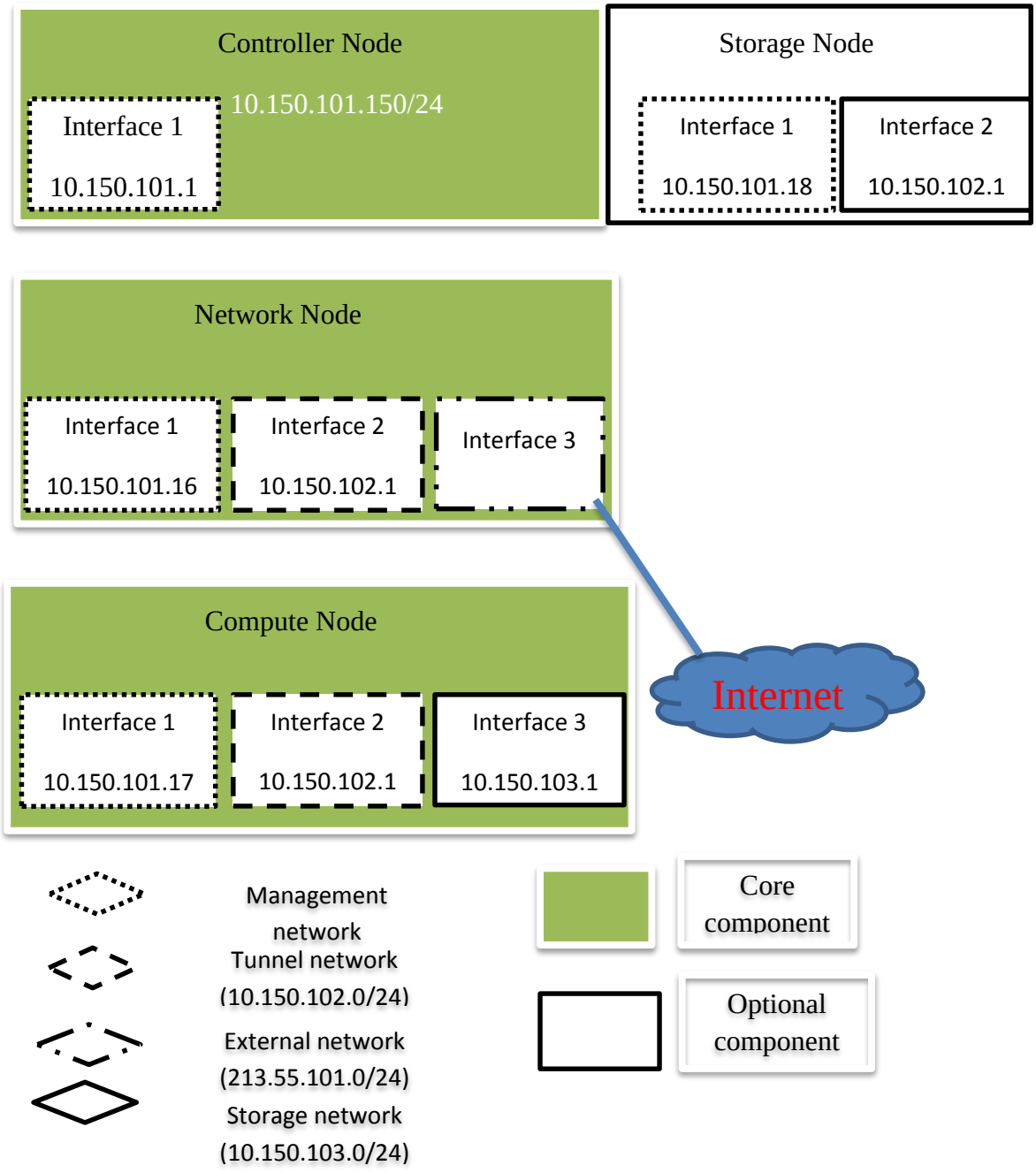

Figure 4-2 Neutron local cloud architecture

Network Configuration

After neutron layout and IP address is decided, the next step is configuring and editing of network, changing the hostname of the computer node. This step is strictly recommended to configure because without applying this step it is impossible to have intercommunication of each node. In the first stage we assign IP address manually then we change the name of the computer node using terminal session. The detail IP address assigned to each node is described in the next table.

Table 4-2 IP address assigned for each computer

Name	Interface (NIC)	IP address	Gateway	Subnet-mask
Controller node	1	10.150.101.150	10.150.101.1	255.255.255.0
Network node	1	10.150.101.160		
	2	10.150.102.160		
	3	Not assigned		
Compute node	1	10.150.101.170	10.150.101.1	
	2	10.150.102.170		255.255.255.0
	3	10.150.103.170		Not assigned
Storage node	1	10.150.101.180		255.255.255.0
	2	10.150.102.180		

Depending on the IP address assigned in the above table, each node has been configured as follows:

Controller node

Configure network

1. The controller node configured the first interface as the management interface and edited manually in the internet connection.

   IP address: 10.150.101
   Network mask: 255.255.255.0
   Default gateway: 10.150.101.1

Configure name resolution

2. Change the hostname of controller node; in our case controller
3. After this step has been finished reboot the system for connection purpose
4. Edit the /etc/hosts file to contain the following:

   Controller
   10.150.101.150 Controller
   Network
   10.150.101.160 Network
   Compute
   10.150.101.170 Compute

Network node

Configure networking

1. Configure the first interface as the management interface:
   IP address: 10.150.101.160
   Network mask: 255.255.255.0
   Default gateway: 10.150.101.1
2. Configure the second interface as the instance tunnels interface:
   IP address: 10.150.102.160
   Network mask: 255.255.255.0

3. The external interface uses a special configuration without an IP address assigned to it.
4. Reboot the system to activate the changes.

To configure name resolution:

1. Set the hostname of the node to network.
2. Edit the /etc/hosts file to contain the following:

```
Network
10.150.101.160      Network
Controller
10.150.101.150      Controller
Compute
10.150.101.170      Compute
```

Compute node

To configure networking:

5. Configure the first interface as the management interface

   IP address: 10.150.101.170

   Network mask: 255.255.255.0

   Default gateway: 10.150.101.1

6. Configure the second interface as the instance tunnels interface:

   IP address: 10.150.102.170

   Network mask: 255.255.255.0

7. Reboot the system to activate the changes.

To configure name resolution:

1. Set the hostname of the node to compute.

2. Edit the /etc/hosts file to contain the following:

```
Compute
10.150.101.170      Compute
Controller
10.150.101.150      Controller
Network
10.150.101.160      Network
```

4.3.Experimental setup details

The experiment was conducted in Ubuntu operating system in which the evaluation was run number of times both in virtual machine of cloud environment. The experiment was executed in number of virtual machines environment. While request and response time was performed, the key performance indicators are gathered for the next evaluations of the result.

Hardware virtualization enables a single physical platform to run multiple operating systems and software stacks. Virtualization creates an abstraction layer between user and physical resource but at the same time it provides the user the illusion of direct interaction with the physical resource. The Virtual Machine Monitor (VMM), also known as the hypervisor, establishes the abstraction layer that encapsulates and isolates each Virtual Machine (VM). The VMM runs on the actual machine and maps physical resources (processing power, memory, storage, and network) to VMs. The Operating System (OS) running inside a VM make use of the virtual resources mapped to the confining VM. Consequently, the physical resources of a machine can be partitioned between multiple VMs.

On virtual machines, one type of application is installed. When local cloud was installed and displayed horizon dashboard, instances run through mechanism of virtualization. The middleware called Xen hypervisor which was installed on hosts of computer and manages VMs. The other virtual machines request application service from the cloud which was installed on the other VMs.

Following are the steps that we took to perform our test in local cloud test environment.
After local cloud was deployed, it requests username and password for authentication and authorization. We got two way of login page. One is demo page and the other one is admin page which is required for administrator only.

Local cloud interface

When the administrator login to page or cloud, there are different resource types assigned for the next creation of virtual machines. From the figure below we can understand resource type like instances used, vCPUs given, RAM used, floating Ips, security groups, volumes used and volumes storage size have been given for creating instances and deploying applications.

Local cloud interface overview

Local cloud deployed supports creating instances and booting instances using ISO images. The instance should be functional if the image boot. Ubuntu iso image was used. Instance name is the name of the new instance created in the cloud. After the instance is successfully launched, it is connected to the instance using remote console to boot the system and run.

Creating an instance on local cloud

We could list with all volumes in the local cloud system. In this list, we could find the volume that is attached to ISO image created instance. Volume is uploaded to glance. The volume has ID of the volume that attached to ISO created instance. When the image is successfully uploaded, it is possible to use the new image to boot instances.

Creating volume on local cloud

To run instances and tenants, configuration of networking has important role. It used for communication of each virtual machine when we access instance image.

Creating network on local cloud

After the network is created, we get network topology which shows the private network created and connected to public network. Router is also created to facilitate flow of data to external network.

CHAPTER FIVE

5. Implementation and Analysis

5.1. Performance parameters and computing resource

To do this research, it is necessary to identify and choose parameters during experimentation and implementation phase. Requirements have been chosen, in order to measure the performance of cloud providers. For instance, specification of instance of virtual machines for selected cloud providers. The resource characteristics for the instance types offered to select clouds have been identified.

We specify computing resource and requirements for testing all clouds with the same application. Table 5.1 indicates the information which is briefly discussed. The price calculations are based on the region in which cloud instance is running. Because, when the distance of data center increase there is an increase of cost on the user since bandwidth has been added to the dedicated virtual machines.

Table 5-1 Computing resource and performance link

Provider name	Instance name	CPU core/s	RAM size (GB)	Hard disk (GB)	Performance
Local cloud	M1.medium	3	5	250	View
Amazon EC2	M1.medium	3	5	250	View
Windows Azure	M1.medium	3	5	250	View

Table 5.3 shows local cloud and other two cloud providers which have been tested using the same computing resource. Performance column has been classified into cost and performance metrics which were discussed in the table 5.2 and table 5.3.

Table 5-2 : Price of instance for each provider

Provider name	Instance name	Price Per Hour
Local cloud	M1.medium	0.062$
Amazon EC2	M1.medium	0.0712$
Windows Azure	M1.medium	0.0701$

Local cloud Performance analysis

Depending on the resource given, we gather different performance. The performance is got from several different cloud providers by the same computing resource except cost is displayed in the following table which is substituted in the implementation by a name called view under the column of performance in the table 5.5.

Table 5-3 : Performance description of local cloud

Provider name	Throughput	Response time	Latency
Local cloud	0.0041	0.05	0.00114
Amazon EC2	0.0034	0.032	0.0032
Windows Azure	0.0023	0.021	0.0026

Requirements used for testing local cloud, Amazon EC2 and Windows Azure

To come over the research work and convert the approaches taken, various requirements are decided. CPU core, RAM size and hard disk are computing resources studied. Instance type, number of users, time to run and type of application are factors used to identify difference providers' performance. This methodology is very important especially for those users who need to know detail performance

of cloud providers' service. So, for different cloud providers, the researchers took the same resource type, similar requirement factors and one type of application.

When testing has been processed, we took number of users based on user base. Since different users of cloud computing come from different locations, our system gives distributive testing. For this reason we categorize our selective testing by location of users.

Table 5-4 Requirements needed to measure cloud provider performance

Provider name	Instance name	No. of users	Time to run	Application type
Local cloud	M1.medium	7	5 minutes	Network bound
Amazon EC2	M1.medium	7	5 minutes	Network bound
Windows Azure	M1.medium	7	5 minutes	Network bound

User interface

According to the proposed algorithm, we implemented STFCC interface. The web page designed solves problems of users by providing interactive interface with any cloud computing. So, anybody can test cloud computing based on testing requirements. The system gives different information about any kinds of cloud computing. For example, any cloud providers which gives infrastructure as a service has been explained with their properties like hypervisor they support, types of operating system, location of data center and deployment model. Our first interface provides interactive system between user and CC. We can get cost estimation for type of virtual machine selected.

Figure 5-1: Cloud provider description

When new cloud provider developed, the system accepts and evaluates performance of virtual machines. After provider performance tested new provider information added to database. Since database refreshed according to set time, any users those come to test cloud will get performance of newly added cloud provider. The following interface requests new provider to full the requirement before join to CID.

Figure 5-2: Register new cloud providers

Testing requirements

For users of cloud, the system gives service of selective testing based on the requirements of the users. The system requests different requirements for the purpose of testing different cloud providers. After all actions are taken, the performance of cloud is displayed by the graph or table form which shows throughput, response time, latency and cost. Every cloud providers' performance data should be collected and stored into CID. To get this, STFCC should be registered for any new cloud provider as discussed in the figure 5-2. Their performance and every requirement and computing resource is measured and analyzed by using VMM. The interface below shows what types of requirement should be included at the time of testing and it is must to add to CID.

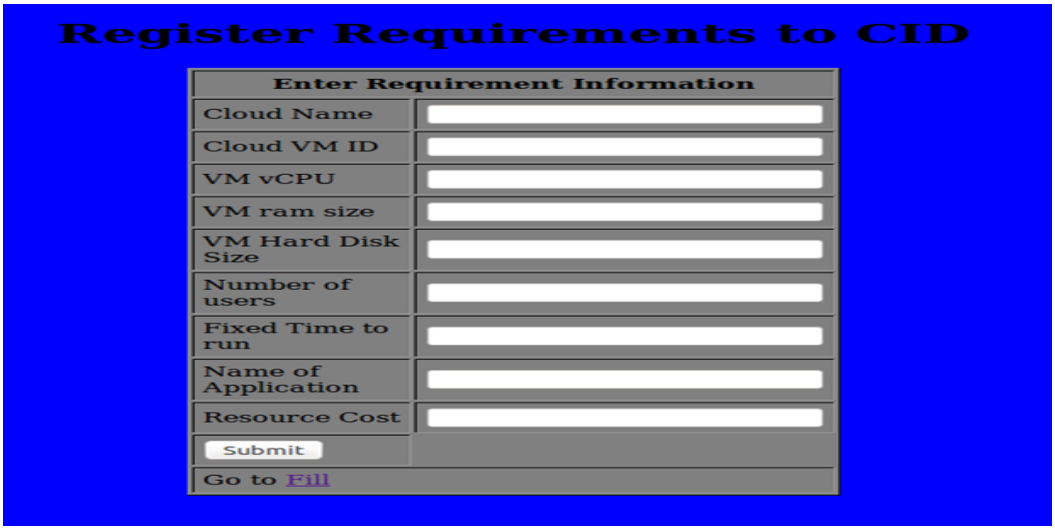

Figure 5-3: Requirements to test cloud providers

Medium sized instances result

In this experiment, m1.medium instances were tested at the same time, having the same compute resource and similar requirements. This helps to investigate the network performance when various providers use similar VMs and also used resource at the same time. We have taken medium sized VMs for testing different provider service performance. Based on the medium sized instances, we has seen performance and cost difference in the table 5.2 and in the table 5.3.

Throughput analysis and results of local cloud

We have local cloud which is deployed in lab test environment. In this case, different test has been taken and got performance metrics. From those all, the analysis shows that throughput is one of them. Number of users sent request to the virtual machines through interface to analyze throughput. So, we run 7 numbers of users to upload and download data. Type of application processed were network traffic to see flow of traffic and how the cloud has been stressed and transfer throughput. The figure 5.3 depicts throughput generated by local cloud based on the requirements given.

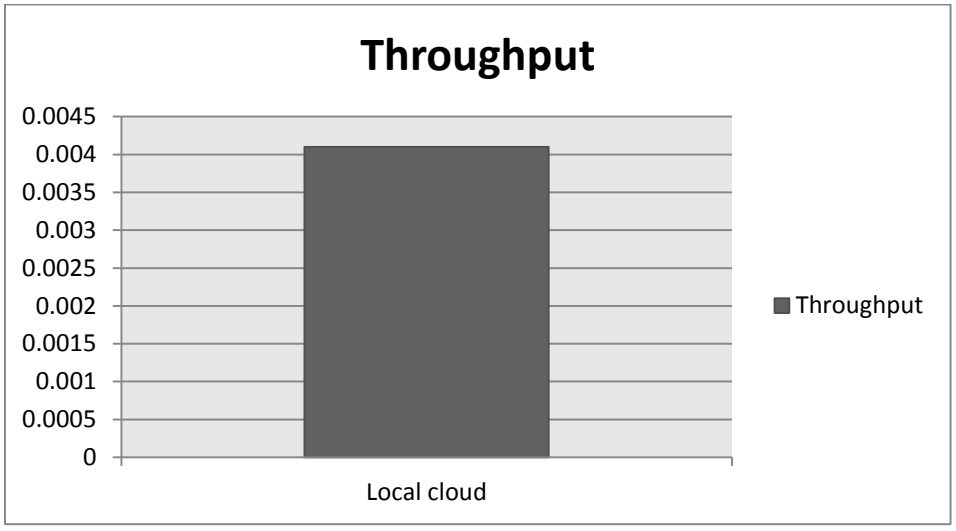

Figure 5-4 Throughput analysis of Local cloud

Response time analysis and results of local cloud

The result of response time that was tested in laboratory environment from local cloud shows that 0.05 ms which was tested for 5 minutes by 7 numbers of users. Application was tested using uploads and downloads of data from virtual machines. The application type was network traffic and up on a test, it indicates there is slow packet flow and the result shows that there is problem on response time when it was compared with Amazon EC2.

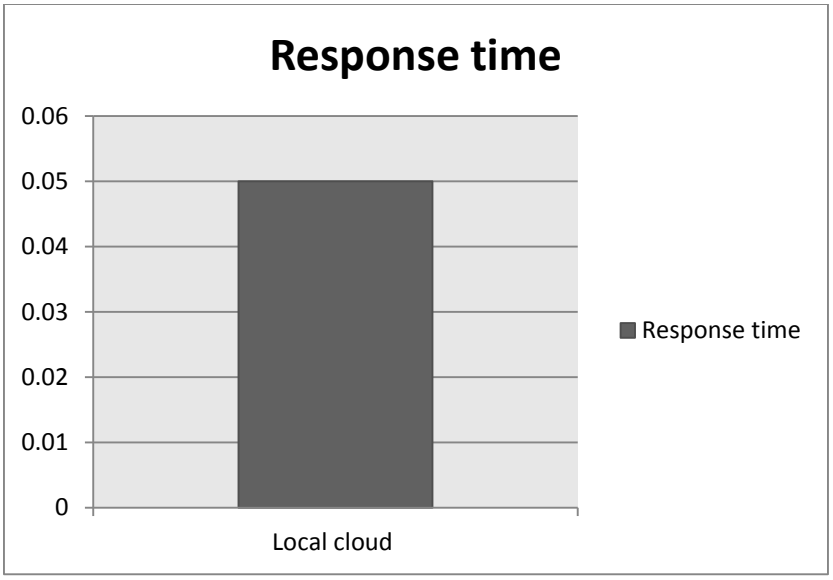

Figure 5-5 Response time of Local cloud

Throughput comparison of local cloud, Amazon and Windows Azure

Throughout has been measured by placing an application on virtual machines. In this study throughput analyzed by deploying local cloud and generating data from database for the purpose of showing sample difference particularly for other cloud. As the figure 5.5 indicates throughput of Local cloud is high when compared to Amazon EC2 and Windows Azure. In the case of local cloud, the performance generated is high. The number of request per time determine throughput of cloud. So, the time against each activity or process were taken and written down to measure and analyze request of workload per given time to conclude throughput of each and every cloud.

On the same computing resource and requirements, workloads were parallel processed. The times against each process was determined and measured by sing and requesting application which stored on virtual machines. Finally throughput has been generated. Figure 5-5 depicts the throughput measured every 5 minutes for 7 clients or users. We analyzed and concluded that the throughput of Local cloud is better than the other cloud environment as shown in Figure 5-5.

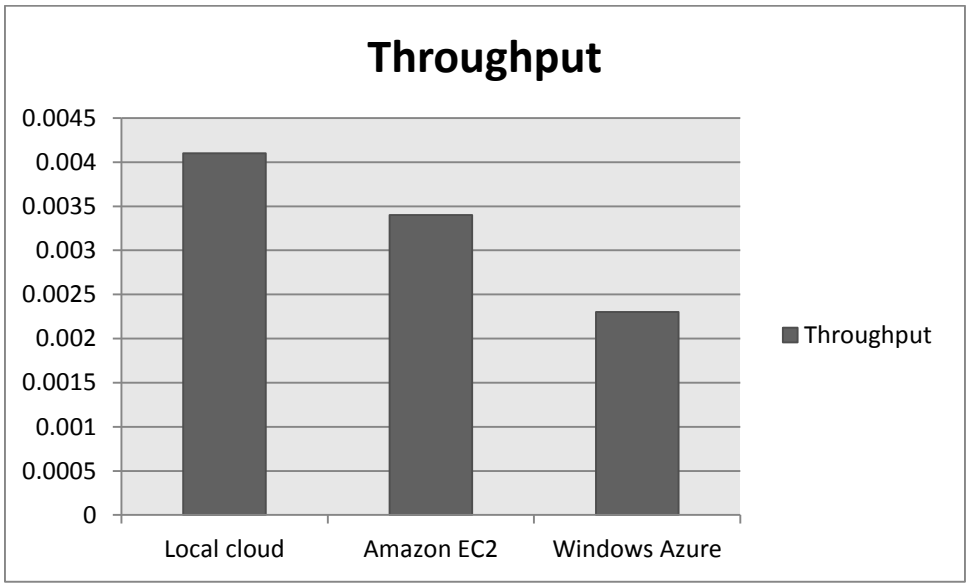

Figure 5-6 Comparison of cloud provider throughput

Response time comparison of local cloud, Amazon and Windows Azure

To see the performance of different clouds, we have local cloud which has been deployed in lab test environment. Windows Azure has been taken for comparison as a sample from database. This comparison indicates that windows Azure takes less time to response for the requested data. The application of network bound was made to run. This means in windows Azure cloud provider, server is not bottlenecked compared to the other two clouds for quick response. Amazon EC2 is fast in response time than local cloud. So, the user can decide which cloud computing is best for them according to their requirements and also depending on the given testing as a service.

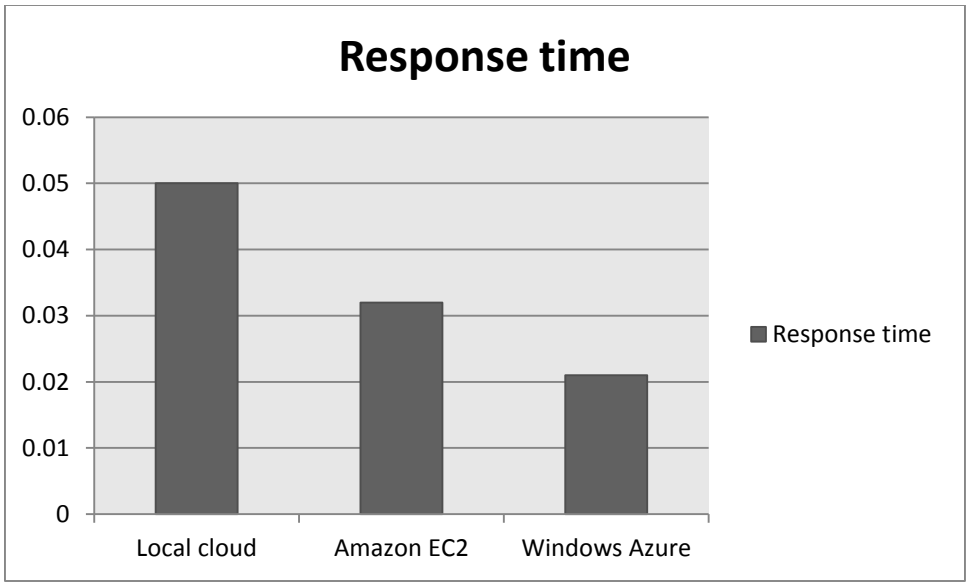

Figure 5-7 Comparison of cloud provider response time

If we compare and contrast the execution times in Tables 5.1 for workload 1-7, the execution time shows difference for the same computing resources for the local cloud, Amazon EC2 and Windows Azure environment. However, the response time was twice as high for the Windows Azure, i.e. it takes 0.021 seconds to process 7 parallel requests in 5 minutes test in the cloud environment. For Amazon EC2, it took 0.032 and 0.05 for Local cloud to do the same thing.

Price comparison of local cloud, Amazon EC2 and Windows Azure

Any users of cloud computing contracts with cloud provider for cost pricing. Providers of cloud disallow accessing the data for corporate gain if there is no agreement and no payment is made for the resources. In our work, local cloud includes dedicated CPU cores, dedicated RAM, hard disk and others. This could be known only by testing the service of cloud providers. The VM configuration tested was m1.medium instance. All details of providers' name, instance name and cost per hour have been discussed in the table 5.3. From the scenario we conclude Local cloud price is lower than Amazon EC2 and Windows Azure.

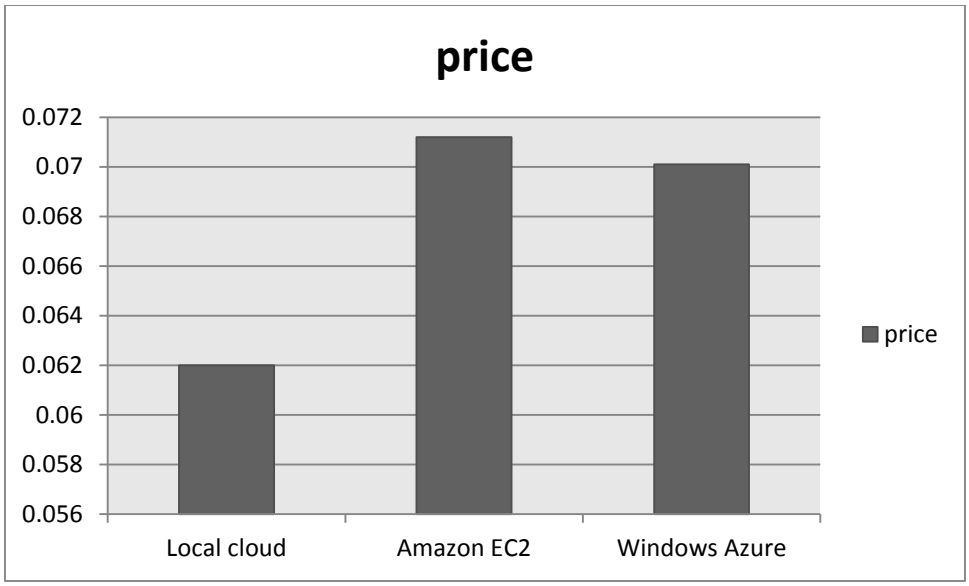

Figure 5-8 Comparison of cloud provider cost price

Latency comparison of Local cloud, Amazon EC2 and Windows Azure

The figure 5-9 illustrates packet delay for VMs on different cloud providers when they were configured with the same computing resource (i.e. CPU core, ram and hard disk). Lower latency shows packets arrived at faster speed to their endpoint. This result is confirmed by the previous throughput discussion which showed local cloud has higher throughput implies lower packet delay or latency.

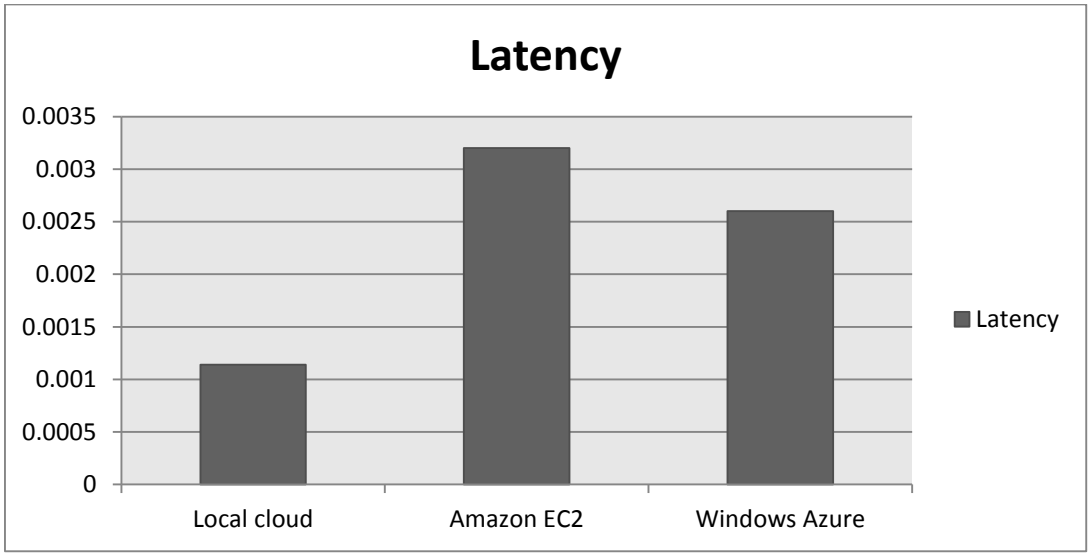

Figure 5-9: Latency comparison of Local cloud Amazon EC2 and Windows Azure

Amazon EC2 and Windows Azure show great packet loss when compared with local cloud. This indicates that if there is high throughput, there is slow response time. Comparing latency has to be related with response time. High response time indicates minimized delay and packet loss.

5.2. Discussions

Some research has been done to test performance of cloud computing. The study brings different approaches and algorithms. Most of approaches are linked to deploying cloud computing on local environment to know and evaluate performance. They took different parameters and requirements. In other research, selective testing for SaaS has been done using different tools. All of the research done focused on testing performance of cloud computing.

Our study differentiate from all of research done by providing selective testing for cloud computing. As explained throughout this paper, our study based on giving testing services for cloud providers in different aspects such as types of cloud deployment model, hypervisor, service type, location of data center and kind operating system supported. Additionally, we test throughput, response time, latency and cost.

Every cloud providers tested before included to CID. According to our approaches and algorithms, new cloud providers tested and evaluated by registering to VVM. Then VMM measures their performance and saves to database. The algorithm developed performs on the same computing resource, the same number of users and requirements. Users are located in different locations. For this reason our step of testing considered distributive testing. The previous testing system is not distributive testing.

CHAPTER SIX

6. Conclusions

Owing to its importance and impact of selective testing and performance evaluations, there is an ongoing research that is in advance in cloud computing. As new innovations arrive, it is equivalent to evaluate new technologies applied on cloud computing environment to enhance the correct understanding about new technology and giving testing of cloud performance status.

This study investigated the possible selective testing for cloud computing. Several steps and procedures are applied to test cloud computing. In the beginning local cloud deployed for prototype and to analyze its performance based on our requirements. On the virtual machines of local cloud we put one file application. After that users permitted to access the application that stored on virtual machines. This is done by the process of request and response. When this activities happen we able to evaluate performance of local cloud. This experiment was carried out to show a performance of cloud, highlighting steps involved in setting up a cloud laboratory environment for testing and evaluations. It gives value to demonstrate the efficiency, benefits and result obtained from cloud technology.

In order to perform evaluation and to get result, the study addressed RQs. To answer the research question, factors have been examined which affect the performance of cloud server. Finally we compare and analyze the descriptive analysis of the results. The evaluation was done using instance of local cloud to test cloud performance by deploying application on local cloud environment. For this purpose we have conducted 7 clients that request and get response. When this type of experiment was applied, we observed PKI (i.e. throughput, response time and latency). Results gathered from the experiment were analyzed and represented in table and graph form.

The algorithm was implemented and tested using designing selective testing interface with single cloud database and deploying local cloud environment by using open source cloud provider. From local cloud lab test, we get real PKI and compare with Amazon EC2 and Window Azure cloud providers. The providers PKI got from gathered information on line, specially computing resource and converting them into algorithm. Using AppManager, Amazon EC2 was tested and response time was also measured. In the case of Amazon performance, we uploaded one file on instance of it and observed what have been going on and analyzed and captured how much time the response time would take to process a response. So, based on the response time, we concluded that response time of Amazon EC2 was higher than local cloud and slower than Windows Azure.

We found that throughput of local cloud and Amazon EC2 by testing on laboratory environment. Window Azure performance was attained by changing the approaches to implementations. All clouds were tested by workload of 1-7 users. This test was done by the interval of 5 minutes based on constant computing resources and requirements. The analyzed result got from both experimentation and implementation. So, data obtained from algorithm and experiment show that throughout of local cloud is higher than Amazon EC2 and Windows Azure. Throughput of Amazon EC2 was slower than local cloud and higher than Windows Azure. It is possible to conclude that local cloud performs and give best performance for users when we test using similar applications and requirements. Response time of cloud was measured with unique computing resources and requirements. From the results displayed, the response time of Windows Azure is higher than Local cloud and Amazon EC2. On top of this the study adds testing cloud providers pricing techniques.

The study was designed and reached algorithmic approaches based on providers' pricing policy and we got that by the same computing resource, it was good to display and measure cloud infrastructure as well as other cost assigning system. Cost data gathered from different sources of cloud providers. The data helps us to compare and contrast Local cloud, Amazon and Window Azure price. Since providers have their own way of cost calculation methods, selective testing gives system of selecting provider payment based on the users that match with their interest. In this study, local cloud cost is lower than Amazon EC2 and Windows Azure. This difference is based on the policy of cloud providers' amendment.

Generally, selective testing for cloud performance in cloud computing is big, bold, interesting and attractive research study based on the result and advantages that give for the cloud users. It is prerequisite for those users, enterprises and organizations going to subscribe their applications in cloud computing. The prerequisite testing clouds may save users from the risk of prospective performance degradation.

Future work

In order to generalize the results we need to perform the same experiment on some cloud providers. Since the data obtained from the experiment presented only three cloud providers, it is difficult to generalize the results to other available cloud computing environments; for example, the performances of other clouds like Rackspace and Opennebula. Rackspace may perform better than the one we deployed and tested.

In this research, unforeseen results have been seen and the questions which were unanswered were popped up. Having experienced the unknown challenges as well as considering the importance of dealing with some problems, it can be suggested further research work on the following areas:

1. Test performance of cloud computing by deploying other cloud provider on physical environment.
2. Test all types of instance then measure and analyze under congested environment.
3. Work on testing as a service, generate other PKI like time to live, jitter, etc.
4. Study on how to optimize performance when you come with performance difference with cloud providers by assigning the same computing resource.

References

[1] Mell, P., and Grance, T., "The NIST definition of cloud computing. National Institute of Standards and Technology," 2011.

[2] DC Plummer, TJ Bittman, Austin, T., Clearley, D. and DM Smith, 12 05 2012. [Online]. [Accessed March 12 2012].

[3] Khatib, V. and Khatibi, E., "Issues on Cloud Computing: A Systematic Review.v," in *International Conference on Computational Techniques and Mobile Computing.*, 2012.

[4] Rajkumar Buyya1,2, Chee Shin Yeo1, and Srikumar Venugopal1, "Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities," in *The 10th IEEE International Conference on High Performance Computing and Communications. IEEE Computer Society.*, Australia, 2008.

[5] Bhaswati Hazarika1, Thoudam Johnson Singh, "Survey Paper on Cloud Computing & Cloud Monitoring: Basics," *SSRG International Journal of Computer Science and Engineering (SSRG-IJCSE) ,* vol. volume 2 , no. issue 1, January 2015.

[6] Faisal Amin and Majid Khan, "Web Server Performance Evaluation in Cloud Computing and Local Environment," Sweden, May 2012.

[7] Jackson, K. (n.d.)., May 2010. [Online]. Available: http://www.xmind.net/share/_embed/kvjacksn/cloud-computing-101/. [Accessed Monday September 2015].

[8] "Understanding The Cloud Computing Stack SaaS, PaaS, IaaS," 2011. [Online]. Available: www.rackspaceclouduniversity.com. [Accessed Friday September 2015].

[9] Srivatsan Jagannathan, "Comparison and Evaluation of Open-source Cloud Management Software," 2012.

[10] M. O'Neill, "Connecting to the cloud, Part 1: Leverage the cloud in applications. IBM.," Retrieved May 26 2010. [Online]. Available: http://www.ibm.com/developerworks/webservices/library/x-cloudpt1/index.html. [Accessed Retrieved April 8, 2015].

[11] T. Merrill, *Cloud Computing: Is your Comany Weighing both Benefits and risks,* Philadelphia, United States: ACE USA, April 2014.

[12] U. Tadahiro, "Selective testing(TaaS) on Clouds," 2013 IEEE Seventh International Symposium on Service-Oriented System Engineering, USA, 2013.

[13] "Software Testing as a Service: Perceptions from Practice," Lappeenranta University of Technology , Finland, 2010.

[14] "Amazon Web Services User Guide: Amazon EC2 Cost Comparison Calculator," 1 February 2010. [Online]. Available: http://aws.amazon.com/economics. [Accessed July 2015].

[15] ProfitBricks with pricing and performance data provided by Cloud Spectator, Cambridge, German, July 2013.

[16] Evangelinos, C. & Hill, C. N, "Cloud Computing for Parallel Scientific HPC Applications: Feasibility of Running Coupled Atmosphere-Ocean Climate Models on," Chicago , 2008.

[17] Zhang, S. Zhang, S. Chen, X. & Wu, S., "Analysis and Research of Cloud Computing System Instance," in *Second International Conference on Future Networks, IEEE*, 2010.

[18] [Online]. Available: http://www.microsoft.com/azure/default.mspx. [Accessed January 2015].

[19] Zhang, S. Zhang, S. Chen, X. & Wu, S., "Analysis and Research of Cloud Computing," in *Second International Conference on Future Networks, 2010 IEEE*, 2010.

[20] Rohit Kamboj*, Anoopa Arya, "Openstack: Open Source Cloud Computing IaaS Platform," *International Journal of Advanced Research in Computer Science and Software Engineering,* vol. 4, no. 5, 2014.

[21] D. U. R. Pol, "Developers Harness Open Source Cloud Management Platforms for Novel Applications," *Urmila R. Pol /(IJCSIT) International Journal of Computer Science and Information Technologies,* vol. 5 , no. 5, 2014.

[22] "Openstack Installation Guide," 7 July 2015. [Online]. Available: docs.openstack.org. [Accessed 22 May 2015].

[23] "Openstack Components," [Online]. Available: http://docs.openstack.org/arch-design/content/openstack-components-arch-storage.html. [Accessed 30 May 2015].

[24] Ralph LaBarge1 and Thomas McGuire2, "CLOUD PENETRATION TESTING," *International Journal on Cloud Computing: Services and Architecture (IJCCSA),* vol. 2, 6 December 2012.

[25] Lakshmi D Kurup, Chandni Chandawalla, Zalak Parekh, Kunjita Sampat, "Comparative Study of Eucalyptus, Open Stack and Nimbus," *International Journal of Soft Computing and Engineering (IJSCE),* Vols. Volume-4, January 2015 2015.

[26] Ishtiaq Ali1 and Natarajan Meghanathan, "VIRTUAL MACHINES AND NETWORKS –INSTALLATION, PERFORMANCE, STUDY,ADVANTAGES AND VIRTUALIZATION OPTIONS," *International Journal of Network Security & Its Applications (IJNSA),* vol. 3, 2011.

[27] Niloofar Khanghahi and Reza Ravanmehr, "Cloud Computing Performance Evaluation: Issues and Challenges," *International Journal on Cloud Computing: Services and Architecture (IJCCSA),* October 2013.

[28] Sinung Suakanto Suhono H Supangkat Suhardi Roberd Saragih, "PERFORMANCE MEASUREMENT OF CLOUD COMPUTING SERVICES," *International Journal on Cloud Computing: Services and Architecture(IJCCSA),Vol.2, No.2, April 2012,* vol. 2, April 2012.

[29] Jerry Gao, K. Manjula, P. Roopa, and E. Sumalatha, "A Cloud-Based TaaS Infrastructure with Tools for SaaS Validation, Performance and Scalability Evaluation," *2012 IEEE 4th International Conference on Cloud Computing Technology and Science,* 2012.

[30] Mumtaz M.Ali AL-Mukhtar, Asraa Abdulrazak Ali Mardan, "Performance Evaluation of Private Clouds Eucalyptus versus CloudStack," *(IJACSA) International Journal of Advanced Computer Science and Applications,* vol. 5, 2014.

[31] Vijayasanthi M1, Satyendra Kumar T2, Jaya Chandra S3, "Cloud Computing: Performance Study in an Eucalyptus Private Cloud," *International Journal of Emerging Technology and Advanced Engineering,* vol. 5, no. 1, January 2015 2015.

[32] Robayet Nasim Andreas J. Kassler, "Deploying OpenStack: Virtual Infrastructure or Dediccated Hardware," in *2014 IEEE 38th Annual International Computers, Software and Applications Conference Workshops*, 2014.

[33] Swapna Addamani, Anirban Basu, "Performance Analysis of Cloud Computing Platform," *International Journal of Applied Information Systems (IJAIS),* vol. 4, no. 4, 4 October 2012.

[34] *Amazon Web Services User Guide: Amazon EC2 Cost Comparison Calculator,* Amazon Web Services, 2010.

[35] "Introduction to Cloud Computing," Office of the Privacy Commissioner of Canada, Canada, 2010.

[36] "openstack," 2015. [Online]. Available: http://www.openstack.org/. [Accessed March].

Appendix

First Page STFCC

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
   pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<style>
   h1 {color:red;
   font-size:1em;}
   body {background-image:
   url("c1.gif");
   }a{color:white;
   font-size:2em;
   url("button.png");}
   </style>
<body>
<h1><blink>SELECTIVE TESTINGFOR CLOUD PERFORMANCE</blink></h1><br>
<a href='Form1.jsp'><img src="button.png"></a>         
<a href='valid.jsp'><img src="l.png"></a>         
<a href='provider.jsp'><img src="p.png"></a>         
<a href='About%20.html'><img src="A.png"></a>
</body>
</html>
```

Administration authentication

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
   pageEncoding="UTF-8"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<script>
function validate(){
var username=document.form.user.value;
var password=document.form.pass.value;
if(username==""){
 alert("Enter Username!");
  return false;
}
if(password=="gar1"){
 alert("Enter Password!");
  return false;
}
return true;
```

```
}
</script>
<form name="form" method="post" action="check.jsp" onsubmit="javascript:return valid();">
<table>
<tr><td>Username:</td><td><input type="text" name="User"></td></tr>
<tr><td>Password:</td><td><input type="password" name="pass"></td></tr>
<tr><td></td><td><input type="submit" value="Login"></td></tr>

</table>
</form>
<body>
<a href="Form1.jsp">Home</a>
</body>
</html>
```

Provider Information

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<style>
    h1 {color:red;
    font-size:1em;}
    body {background-image:
    url("images.jpeg");
    }a{color:white;
    font-size:2em;}
    </style>
<body>
<h1><blink>LISTS OF CLOUD COMPUTING THAT PROVIDE DIFFERENT SERVICES</blink></h1><br>
<select name="forma" onchange="location = this.options[this.selectedIndex].value;">
 <option value="OS.jsp">List Providers Service Type</option>
 <option value="OS.jsp">OS</option>
 <option value="frame2.jsp">Providers</option>
 <option value="paas.jsp">PaaS Providers</option>
 <option value="iaas.jsp">IaaS Providers</option>

</select>
<script type="text/javascript">
<!--
function go(){
location=
document.mycombo.example.
options[document.mycombo.example.selectedIndex].value
}
//-->
</script>
  <input type="button" name="test" value="Go!" onClick="go()">
```

```
<a href='serviceType.jsp'>Provider</a>

<body>

</body>
</html>
```

Cloud Computing Providers Description

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
   pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<frameset cols="25%,25%,25%,25%">
<frameset rows="*,200">
<frame src="saas.jsp">
<frame src="serverlocation.jsp">

</frameset>
<frame src="servicemodel.jsp">
<frame src="depmodel.jsp">
<frame src="serveros.jsp">
</frameset>
<body>

</body>
</html>
```

Cloud Providers Service Type

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
   pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<style>
   h1 {color:red;
   font-size:1em;}
   body {background-image:
   url("c1.gif");
   }a{color:white;
   font-size:2em;
   url("button.png");}
   </style>
<body>
<h1><blink>Service Model</blink></h1><br>
```

```
        .Infrastructure as a service (IaaS)


</body>
</html>
```

Deployment model information

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<style>
    h1 {color:red;
    font-size:1em;}
    body {background-image:
    url("c1.gif");
    }a{color:white;
    font-size:2em;
    url("button.png");}
    </style>
<body>
<h1><blink>Deployment Model</blink></h1><br>

        .Public<br>
        .Private<br>
        .Hybrid


</body>
</html>
```

Cloud Providers Operating system Type

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<body>

Choose operating system <br><br><br>

            <form name="form" method="get" action="servlet/PrintResultServlet"></form>
            <select name="events">
```

```
            <option value="0" selected >Operating System</option>
            <option value="1">Ubuntu</option>
             <option value="2">Windows</option>

</select>
</body>
</html>
```

Cloud Providers Data Center Location

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
   pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<style>
   h1 {color:red;
   font-size:1em;}
   body {background-image:
   url("c1.gif");
   }a{color:white;
   font-size:2em;
   url("button.png");}
   </style>
<body>
<h1><blink>Server Locations</blink></h1><br>

        .Africa<br><br>
        .Asia<br><br>
        .Antartica<br><br>
        .Australia<br>
        .Europe<br>
        .North America<br>
        .South America<br>


</body>
</html>
```

Registration of New Cloud Through STFCC

```
<%@ page import ="java.sql.*" %>
<%@page import="java.sql.*,java.util.*"%>
<%
   String id = request.getParameter("id");
   String name = request.getParameter("name");
   String location = request.getParameter("location");
   String email = request.getParameter("email");
   String country = request.getParameter("country");
   String cloudAPI = request.getParameter("cloudAPI");
   Class.forName("com.mysql.jdbc.Driver");
   Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/DBname", "root", "admin");
```

```
    Statement st = con.createStatement();
    //ResultSet rs;
    int i = st.executeUpdate("insert into newCloud(id, name, location, email, country, cloudAPI)
values('"+id+"','"+name+"','"+location+"','"+email+"','"+country+"','"+cloudAPI+"')");
    if (i > 0) {
        //session.setAttribute("userid", user);
        response.sendRedirect("welcomee.jsp");
       // out.print("Registration Successfull!"+"<a href='index.jsp'>Go to Login</a>");
    } else {
        response.sendRedirect("update.jsp");
    }
%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Registration</title>
    </head>
    <style>
    h1 {color:red;
    font-size:2em;}
    body {background-image:
    url("cloud.gif");
    }
    </style>
    <body bgcolor="blue">
    <h1 align="center">Register New Cloud Provider</h1>
        <form method="post" action="registration.jsp">
            <center>
            <table border="1" width="30%" cellpadding="5" bgcolor="green">
                <thead>
                    <tr>
                        <th colspan="2">Enter Information Here</th>
                    </tr>
                </thead>
                <tbody>
                    <tr>
                        <td>Cloud ID</td>
                        <td><input type="text" name="id" value="" /></td>
                    </tr>
                    <tr>
                        <td>Name of Cloud</td>
                        <td><input type="text" name="name" value="" /></td>
                    </tr>
                    <tr>
                        <td>Cloud Data Center</td>
                        <td><input type="text" name="location" value="" /></td>
                    </tr>
                    <tr>
                        <td>Providers Email</td>
                        <td><input type="text" name="email" value="" /></td>
                    </tr>
                     <tr>
                        <td>Country</td>
```

```
            <td><input type="text" name="country" value="" /></td>

          </tr>
          <tr>
            <td>API of Cloud</td>
            <td><input type="text" name="cloudAPI" value="" /></td>

          </tr>
          <tr>
            <td><input type="submit" value="Submit" /></td>
            <td><input type="reset" value="Reset" /></td>
          </tr>
          <tr>
            <td colspan="2">Already registered!! <a href="update.jsp">Login Here</a></td>
          </tr>
        </tbody>
      </table>
      </center>
    </form>
  </body>
</html>
```

Computing Resources and Requirements

```
<%@ page import ="java.sql.*" %>
<%@page import="java.sql.*,java.util.*"%>
<%
   String pname = request.getParameter("pname");
   String insname = request.getParameter("insname");
   String cpucore = request.getParameter("cpucore");
   String ram = request.getParameter("ram");
   String disk = request.getParameter("disk");
   String nuser = request.getParameter("nuser");
   String time = request.getParameter("time");
   String appn = request.getParameter("appn");
   String cost = request.getParameter("cost");
   Class.forName("com.mysql.jdbc.Driver");
   Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/DBname", "root", "admin");
   Statement st = con.createStatement();
   //ResultSet rs;
   int i = st.executeUpdate("insert into cloud1(pname, insname, cpucore, ram, disk, nuser, time, appn, cost)
values('"+pname+"','"+insname+"','"+cpucore+"','"+ram+"','"+disk+"','"+nuser+"','"+time+"','"+appn+"','"+cost+"')");
   if (i > 0) {
      //session.setAttribute("userid", user);
      response.sendRedirect("welc.jsp");
     // out.print("Registration Successfull!"+"<a href='index.jsp'>Go to Login</a>");
   } else {
      response.sendRedirect("regperf.jsp");
   }
%>
</body>
</html>
```

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Registration</title>
   </head>
   <style>
   h1 {color:blue;
   font-size:2em;}
   body {background-image:
   url("p.gif");
   }
   </style>
   <body bgcolor="gray">
   <h1 align="center">Provide your Performance to Cloud Center</h1>
      <form method="post" action="insper.jsp">
         <center>
         <table border="1" width="30%" cellpadding="5" bgcolor="gray">
            <thead>
               <tr>
                  <th colspan="2">Enter Requirement Information</th>
               </tr>
            </thead>
            <tbody>
               <tr>
                  <td>Cloud Name</td>
                  <td><input type="text" name="pname" value="" /></td>
               </tr>
               <tr>
                  <td>Cloud VM Name</td>
                  <td><input type="text" name="insname" value="" /></td>
               </tr>
               <tr>
                  <td>VM cpu core/s</td>
                  <td><input type="text" name="cpucore" value="" /></td>
               </tr>
               <tr>
                  <td>VM ram size</td>
                  <td><input type="text" name="ram" value="" /></td>
               </tr>
                <tr>
                  <td>VM Hard Disk Size</td>
                  <td><input type="text" name="disk" value="" /></td>

               </tr>
               <tr>
                  <td>Number of user run</td>
                  <td><input type="text" name="nuser" value="" /></td>
                  </tr>
               <tr>
                 <td>Fixed Time to run</td>
                  <td><input type="text" name="time" value="" /></td>
                 </tr>
               <tr>
                  <td>Name of Application</td>
```

```
                <td><input type="text" name="appn" value="" /></td>
                </tr>
              <tr>
                <td>Resource Cost</td>
                <td><input type="text" name="cost" value="" /></td>
              </tr>
              <tr>
                <td><input type="submit" value="Submit" /></td>

              </tr>
              <tr>
                <td colspan="2">Go to <a href="welc.jsp">Fill</a></td>
              </tr>
            </tbody>
          </table>
          </center>
        </form>
    </body>
</html>
```

Cloud Performance Key Indicators

```
<%@ page import ="java.sql.*" %>
<%@page import="java.sql.*,java.util.*"%>
<%
    String throughput = request.getParameter("throughput");
    String responsetime = request.getParameter("responsetime");
    String latency = request.getParameter("latency");
    String jitter = request.getParameter("jitter");
    String timetolive = request.getParameter("timetolive");
    String delay = request.getParameter("delay");
    String cost = request.getParameter("cost");
    Class.forName("com.mysql.jdbc.Driver");
    Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/DBname", "root", "admin");
    Statement st = con.createStatement();
    //ResultSet rs;
    int i = st.executeUpdate("insert into perf(throughput, responsetime, latency, jitter, timetolive, delay, cost)
values('"+throughput+"','"+responsetime+"','"+latency+"','"+jitter+"','"+timetolive+"','"+delay+"','"+cost+"')");
    if (i > 0) {
        //session.setAttribute("userid", user);
        response.sendRedirect("welcomee.jsp");
       // out.print("Registration Successfull!"+"<a href='index.jsp'>Go to Login</a>");
    } else {
        response.sendRedirect("update.jsp");
    }
%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>Registration</title>
    </head>
    <style>
```

```
h1 {color:blue;
font-size:2em;}
body {background-image:
url("p.gif");
}
</style>
<body bgcolor="gray">
<h1 align="center">Cloud Provider Information</h1>
   <form method="post" action="perf.jsp">
     <center>

      <table border="1" width="30%" cellpadding="5" bgcolor="gray">
         <thead>
            <tr>
               <th colspan="2">Enter Information Here</th>
            </tr>
         </thead>
         <tbody>
            <tr>
               <td>Throughput</td>
               <td><input type="text" name="throughput" value="" /></td>
            </tr>
            <tr>
               <td>Response Time</td>
               <td><input type="text" name="responsetime" value="" /></td>
            </tr>
            <tr>
               <td>Latency</td>
               <td><input type="text" name="latency" value="" /></td>
            </tr>
            <tr>
               <td>Jitter</td>
               <td><input type="text" name="jitter" value="" /></td>
            </tr>
             <tr>
               <td>Time to Live</td>
               <td><input type="text" name="timetolive" value="" /></td>

            </tr>
            <tr>
               <td>Delay</td>
               <td><input type="text" name="delay" value="" /></td>
               </tr>
            <tr>
              <td>Cost</td>
               <td><input type="text" name="cost" value="" /></td>
              </tr>
               <tr>
              <td><input type="submit" value="Submit" /></td>
              </tr>
              <tr>
               <td colspan="2">Go to <a href="welk.jsp">Fill</a></td>
            </tr>
         </tbody>
      </table>  </center>
```

```
      </form>
   </body>
</html>
```

Performance Key Indicators implementation code

```
<%@ page import="java.io.*,java.util.*,java.sql.*"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql"%>

<html>
<head>
<title>SELECT Operation</title>
</head>
<body>

<sql:setDataSource var="snapshot" driver="com.mysql.jdbc.Driver"
    url="jdbc:mysql://localhost/DBname"
    user="root"  password="admin"/>

<sql:query dataSource="${snapshot}" var="result">
SELECT * from cloud1;
</sql:query>
 <h1>Cloud Performance Description</h1>
    <table border="1" width="60%">
<tr>
 <th>Provider Name</th>
  <th>Instance Name</th>
   <th>CPU in Core/s</th>
   <th>RAM(GB)</th>
   <th>Hard Disk(GB)</th>
    <th>No of Users</th>
    <th>Time</th>
     <th>Type of Application</th>
      <th>Cost</th>
    <th>Performance</th>
</tr>
<c:forEach var="row" items="${result.rows}">
<tr>
   <td><c:out value="${row.pname}"/></td>
   <td><c:out value="${row.insname}"/></td>
   <td><c:out value="${row.cpucore}"/></td>
   <td><c:out value="${row.ram}"/></td>
  <td><c:out value="${row.disk}"/>
  <td><c:out value="${row.nuser}"/>
  <td><c:out value="${row.time}"/>
  <td><c:out value="${row.appn}"/>
  <td><c:out value="${row.cost}"/>
  <td colspan="2"><a href="performance.jsp">Info</a></td></tr></td></c:forEach></table>
```

Declaration

I, the undersigned, declare that this thesis is my original work and has not been presented for a degree in any other work, and that all source of materials used for the thesis have been duly acknowledged.

Declared by:

Name: Garamu Tilahun

Signature: _____

Date: _____

Confirmed by advisor:

Name: Frezewd Lema (PhD)

Signature: _____

Date: _____

Place and date of submission: Adama, Feb, 2016