ADDIS ABABA UNIVERSITY

COLLEGE OF NATURAL SCIENCE

Service Level Agreement Based Job Scheduling Model for Cloud Computing Environment

Tilksew Shiferaw

A Thesis Submitted to the Department of Computer Science in Partial Fulfillment for the Degree of Master of Science in Computer Science

Addis Ababa, Ethiopia

7 October 2020

Addis Ababa University

College of Natural Sciences

Tilksew Shiferaw

Advisor: Ayalew Belay (PhD)

This is to certify that the Thesis prepared by Tilksew Shiferaw entitled "*Service Level Agreement Based Job Scheduling Model for Cloud Computing Environment* ". And it is submitted in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science in Network and Security complies with the regulations of the university and meets the accepted standards with respect to originality and quality.

Name and Signature of the Examining Board

1. Advisor: Ayalew Belay (PhD)_____
2. Examiner: Solomon Atinafu (PhD)_____
3. Examiner: Minale Ashagirie (PhD) _____

Abstract

Cloud computing is the most significant and an Internet based computing, providing on-demand services through the Internet such as servers, storage disk, different platforms and applications to any business level company or organization. As cloud computing gives services to users, users can get services from the provider using scheduling process and cloud consumers' needs can vary from organization to organization and time to time. Consequently, this research work identified host utilization and user satisfaction in terms of resource usage are the major problems of job schedulers in the cloud. Thus, we articulated a new scheduling called service level agreement based job scheduling for cloud computing environment.

This study mainly uses the job length and host utilization as parameters. Furthermore, host monitor, job organizer, virtual machine selector and virtual machine placement are the main components of the study. The component job organizer is used to sort jobs based on length in ascending order. Host monitor is used for detecting host whether it is over-loaded or not and it helps to know when the virtual machine is migrated or not. Virtual machine selector selects the virtual machine from the source host to be migrated to the other host by considering a virtual machine, which has minimum migration time. Virtual machine placement accept the input from the virtual machine selector module and it place the virtual machine to the under loaded destination host based on the threshold value set in service level agreement. The implementation of this research is used cloudsim toolkit for executing simulation of result.

Keywords: Cloud computing, job scheduling, service level agreement

Acknowledgment

To a great degree, all praises to Almighty God for giving me the power to accomplish this thesis. I could not finish this work without his blessing. I would like to express my heartfelt gratitude to my advisor Dr. Ayalew Belay who presided over this thesis and in that line improved it significantly. Many thanks to my close friends, Yihnewu Gebru, Tilksew Shimles, Mulat Getaneh, Yared Dessalegn, Bekel Worku, Yitayal Kassie and Mesel Tesfu for stimulating discussions and good memories shared together. Also, I would like to express my profound gratitude to my beloved family, sisters, brothers, who expressed their encouragement and love. Finally, I appreciate the financial support of Addis Ababa University that funded for processing of the research work.

Table of Contents

List of figures

List of tables

List of Algorithms

Acronyms and Abbreviation

BW	Bandwidth
CSC	Cloud service customer
CSP	Cloud service provider
HU	Host Utilization
IaaS	Infrastructure as a service
IT	Information Technology
I/O	Input/ output
MMT	Minimum migration time
OS	Operating system
PaaS	Platform as a service
PDM	Performance degradation due to migrations
QoS	Quality of service
SaaS	Software as a service
SLA	Service level agreement
SLO	Service level objective
SLATAH	Service level agreement Time per Active Host
SLAV	Service level agreement violation
Vm	Virtual machine

Chapter One: Introduction

1. Background

The term computing refers to any activity that uses computers, which can manage, process and communicate information. It includes the development of both hardware and software. Today computer engineering, software engineering, computer science and information system are the major computing disciplines. Computing models are further divided into three types called cluster computing, grid computing and cloud computing [1].

Cluster computing is a collection of parallel or distributed computers, that contains a collection of inter-connected stand-alone computers working together as a single integrated computing resource [2].

Grid computing [1] is a form of parallel and distributed system that can used for sharing, selection and aggregation of geographically distributed independent resources dynamically at runtime depending on quality of service requirements.

Cloud computing [3] is also part of parallel and distributed system, which refers to the applications and services that run in distributed network using virtual resources that can be accessed through the Internet protocol and network standard. It is recognized by the concept that the resources are virtual, extensive and the details of the physical systems on which software runs are abstracted from the user. Cloud computing provides the delivery of software, infrastructure and storage services over the Internet.

In cloud computing there is a concept of deployment model and service model. The deployment model refers to where the cloud is located and for what purpose it is applied. The deployment model is categorized in to four parts namely public cloud, private cloud, hybrid cloud and community cloud. Whereas, the service model describes that the service provider is offering services to the user through the Internet. The most known service model of cloud computing is software as a service (SaaS), platform as a service (PaaS) and infrastructure as a service (IaaS). SaaS provides the software to the user, so that user do not need to install the software on their own machines and they can use the software directly from the cloud. PaaS provides the platform to the clients so that the clients can make their applications on the platform. IaaS provides infrastructure to cloud users for various purpose like, storage system and computation resources [4].

In cloud computing, there are three kinds of parties. Those are cloud service consumer or user that accesses services from a cloud provider, cloud service provider that give services to user and cloud network, which is the Internet that can be considered as the transmission media of the cloud [5].

Resource management and scheduling are the key technological concept for cloud computing that plays an important role in effective management and access of cloud resource. In order to access cloud computing services, scheduling is one of the most important key points. To do that, various kinds of job scheduling are used to access the cloud resources. Job scheduling refers to dispatching the computing tasks to resource pooling between different resources users according to certain rules or Service Level Agreements (SLA) of resource use under a given cloud circumstances. There are different types of resources and various customer needs and SLA between cloud parties in cloud environment. Due to this reason, there is not a uniform standard for job scheduling. Because job scheduling is done based on the necessity or SLA of users and providers who has a set of jobs to execute [6]. The SLA is an agreement made between customers and providers. It is a key aspect when the user or service customer delegates their task to the cloud service provider. SLA between cloud providers and the cloud users are used to assure Quality of Service (QoS) which is one of the big issues that resist organization from availing cloud resources [7].

Different SLA has the same meaning but their syntax may be different or similar according to the requirements of users. Because, SLAs in cloud consists of different specification regarding to security, pricing, performance, resource utilization, customer level satisfaction etc. [8]. Within cloud computing environment, most cloud providers support services under constraints of SLA. The SLAs are composed from different QoS and rules promised by the provider to customers. Due to this, modeling job scheduling based on SLA is more advantageous to achieve high level of user satisfaction and better resource utilization. Therefore, in this thesis, we could model SLA based job scheduling for cloud computing environment.

1.1 Motivation

The rapid growth of cloud computing services and the increasing number of user demands on resources of clouds have led to the development and popularity of cloud networks. The cloud

environment provides much amount of resource to users, so the resources are used on-demand. In cloud computing, cloud resource providers wish to gain much profits from their own resources [9].

As cloud computing give services to users, users can get services from the provider using scheduling process. There are various scheduling type in cloud computing, which contains different QoS parameters. However, when customers want to access cloud services, they need to minimize their costs and achieve their required efficiency [9]. But cloud customers' needs may vary from organization to organization and from time to time. To address this challenging issue, SLA is one factor that can be considered in addressing the issues mentioned above. Due to this reason, we are motivated to conduct a research on job scheduling with respect to SLAs requirement.

1.2 Statement of the problem

Cloud computing is the most significant and an internet based computing, providing on-demand services through the Internet such as servers, storage disk, different platforms and applications to any business level company or organization. The c1oud computing services are "pay based on usage" based on the agreement between service provider and customers. This can make the cloud computing become an attractive for customers. Cloud computing is one of the user-oriented technology in which the user faces a pool of virtualized computer resource through job scheduling. Job scheduling is one of the most important issues in a cloud system in order to allocating certain jobs at particular time and particular resource between various resource users depending on certain rules of resource use under a given cloud features [6].

As customers delegate their tasks to cloud providers, SLA between customers and providers is the main concern. Because of the dynamic nature of the cloud, continuous monitoring on QoS attributes is necessary to enforce SLA. In addition to this, cloud users want to get better services from the cloud provider as per the agreement [10] and sharing of limited hardware resources such as memory, cache and CPU cycles leads to the application performance degradation. To fulfill this, several researchers conducted researches on job scheduling based on SLA on the area of cloud computing [11 - 16]. However, all of them are focus on some specific scheduling parameters like storage, makespan, geography, time, fault tolerance,

completion time, resources utilization, virtual machine bandwidth, and RAM capacity. The researchers do not consider job scheduling based on SLA using priority of job by length. They also, far apart from considering migration process of virtual machine depending on resource utilizations of host.

Depending on these situations, this thesis is aimed to achieve customer satisfaction for the case of resource utilization by detecting over-loaded and under-loaded host and by prioritizing jobs using job length. Accordingly, this thesis aim is to design and model job scheduling using SLA as the main objective of the scheduler to improve customer satisfaction, utilization of resource and relationship between service provider and service consumer.

1.3 Objective

General objective

The general objective of this research is to design and develop job scheduling model on service level agreement for cloud computing.

Specific objectives

- ✓ Conduct a review on services and architecture of cloud computing.
- ✓ Conduct a survey on the existing job scheduling algorithms of cloud computing.
- ✓ Identifying QoS parameters in relation to SLA in cloud computing.
- ✓ Design job scheduling algorithm by taking SLA as an objective of the development.
- ✓ Develop prototype of the model and test the performance using cloud environment simulator.

1.4 Methods

To achieve the objective of this research we will conduct the following methods.

Literature review

In order to get detail understanding of cloud computing services, infrastructures, models literature reviews will be conducted. In the literature review, we will also identify and analyze job scheduling algorithms and useful approaches that can help to solve the identified problem effectively.

Modelling proposed system

From the literature review and analysis of the result of current job scheduling algorithms and QoS in relation to SLA, the proposed solution will be modelled.

Development and evaluation

The prototype for the proposed solution model will be developed using Java programming language and the performance of the prototype will be tested using an appropriate cloud environment simulator called cloudsim.

1.5 Scope and limitations

The main focus of this study is designing, modeling and prototype development of cloud computing job scheduling by taking SLA as an objective. However, this research work does not study or engage on scalability, fault tolerance and security process.

1.6 Application of results

The result of this study could have a contribution to achieving a high level of user satisfaction for clients in the cloud environment by improving high resource utilization. From the organization perspective, it can help the organization to be profitable because of providing better services and capable of leading in a better way of overall system performance. It could be enable the cloud service provider to have a good relationship with their customers. For the research community, it would help other researchers to conduct research on job scheduling algorithm by taking the result of this research work.

1.7 Thesis organization

The rest of this thesis is organized into five major Chapters. Chapter two gives general background on cloud computing like its characteristics, models and virtualization and migration process. It will reviews job scheduling and SLA in cloud environment. In Chapter three, the major scheduling algorithms developed based on SLA for cloud computing are reviewed.

Chapter four will present the proposed system for modeling SLA based job scheduling. In this chapter the architecture of the proposed model and its components are describe in detail. As well as, the working principle of components and their detail description is put precisely.

Chapter five will describe the prototype implementation of the proposed model. Tools used to develop the prototype as well as their significance specific to this prototype development are discussed.

The last section of this thesis is Chapter six. In this Chapter, we will summarize the major concepts raised in this research work. In addition, contributions of the research and future works that could be done to improve the performance of the proposed model are briefly described.

Chapter Two: Literature Review

This Chapter deals about the review of general important concepts concerning this research work. It provides a detailed description of cloud computing from different researchers' views. It also discusses various models and architectures of cloud computing. The part of scheduling within cloud computing environment is discussed and services level agreement in cloud also addressed.

2.1 Cloud computing

Cloud computing is fascinating and an increased number of applications, which used to run on remote datacenters. Cloud computing is a type of computing paradigm, which lots of researchers defined it from different application aspects. Among the various definitions, the followings are some of them.

Cloud computing can assign remote services with a user's data, computation and software. It is a model, which provided on-demand access to shared pool computing resources like servers, storage, networks, applications and services [17].

Cloud computing uses a network of non-local servers hosted on the Internet to store, oversee and handle information, instead of a nearby server. Cloud computing provides services by on-demand method and it is highly demanded service. Since, it has advantages like high computing power, less cost of services, high performance, scalability, reliability, accessibility as well as availability [18].

Cloud computing is an internet-based computing technique and becoming a popular option for renting computing and storage infrastructure services. This can helps for remote platform building and customization for business processes and for renting business applications as a whole. This novel technique can integrate, optimizes and offers computing ability, aiming to simplify the clients computing jobs by the way of renting resources and services [19].

Cloud computing is the model for enabling convenient, on-demand network access to shared configurable resources that can be delivered and discharged quickly with the service provider [3].

Cloud computing has its own components consisting of a wide range of services that can be used by users throughout the internet [20]. One part of its component is client. Clients include

computer users, laptops, tablets, mobile phones etc. The second cloud component is data centres. Datacenter is the set of servers that the applications are hosted. The third component is distributed server. Distributed server is a server, which resides none locally, which is geographically far.

2.2 Necessary of cloud computing

Cloud computing has its own necessity [21] for computing technology as mentioned below.

- ✓ **Availability of resources**

The cloud provides several and varieties of services on the network. The services of cloud are accessed based on the IT needs of a company. Since, the hardware and software resources in cloud computing are available in cloud service providers.

- ✓ **Providing flexibility IT services**

Under cloud computing, the services are provided on monthly basis or on the consumption of resources and the companies can access the services quickly based on their IT needs.

- ✓ **Ability of refreshing an aging infrastructure with less prices**

Currently the companies are virtualizing their critical applications. For virtualizing, the companies use the virtual machines associated with applications to execute on powerful servers and the cloud computing provides the facility without buying a new server.

- ✓ **Economically supporting for new IT services and many users**

Cloud computing provides the facility to forward the applications to customer's infrastructure and save the expenses for datacenter.

2.3 Cloud computing characteristics

Cloud computing can integrate with a number of existing technologies that have been applied in grid computing, utility computing and internet of things. Cloud computing is an unprecedented paradigm for hosting and delivering resources by providing on-demand services and have its own common characteristics [3].

On-demand self-service: It is the primary characteristic of the cloud. Cloud services such as, storage, network access, server time, web applications etc., can be allocated automatically as required by the consumers without any human interaction [22] .

Broad network access: It should be noted that cloud capabilities are available on the network and are accessible through standard mechanisms that use different client platforms such as mobile phones and tablets [22].

Resource pooling: The supplier's resources are pooled to support several customers using a multi-tenant model, by dynamically assigning and reassigning different physical and virtual resources according to customer demand. Customers use multi-tenant models to optimize a resource provider for the customer. It dynamically allocates various physical and virtual resources and defines them according to the demand of the customer [3].

Rapid elasticity: Describes scalable provisioning or the capability to provide scalable cloud computing services. It refers to the capabilities that can be elastically provisioned and released scale rapidly outward and inward commensurate with demand. Resources in cloud computing can be monitored, controlled and reported that provides transparency to suppliers and users about the use of the service [3].

2.4 Architecture of cloud computing

To analyze cloud computing issues, it is important to understand cloud architecture. Depend on National Institute of Standards and Technology (NIST) [3] the reference architecture of cloud computing is described based on the major actors, their activities and functions in the environment of cloud as shown in Figure 2.1.

Cloud customer: A person or an entity that maintains a business relationship and uses cloud provider services. Cloud consumer browses a cloud provider's service catalogue asks for the appropriate service, establishes cloud provider service contracts and uses the service. Cloud consumers need SLAs to determine that a cloud provider meets the technical performance requirements. Cloud providers may also list the collection of SLAs like, restrictions and responsibilities that cloud consumers must recognize.

Cloud provider: An individual, organization, or an entity that is responsible for delivering a service to interested parties. Cloud provider acquires and maintains the computing infrastructure required to deliver the services. It also operates the software delivering service and arranges for cloud services to be provided via network connectivity to cloud clients.

Cloud auditor: Cloud auditor is a party that can perform an independent examination of cloud service controls with the intent to express an opinion thereon. Audits are conducted by

evaluating objective proof to ensure adherence to requirements. A cloud auditor may evaluate a cloud provider's services in terms of security measures, impacts on privacy, efficiency and so on.

Cloud broker: A party that can conduct independent assessment of cloud services, information system operations, performance and security of the cloud implementation. Cloud consumers can request cloud services from a cloud broker, rather than directly contact to the cloud provider. A cloud broker is an entity that manages the use, performance and delivery of cloud services and negotiates relationships between cloud providers and cloud consumers.

Cloud carrier: A medium that provides networks and transport the cloud services from cloud providers to clients. Cloud carriers provide access to consumers through the network, telecommunication and other access devices. For example, cloud consumers can obtain cloud services through network access devices, such as computers, laptops, mobile phones etc.

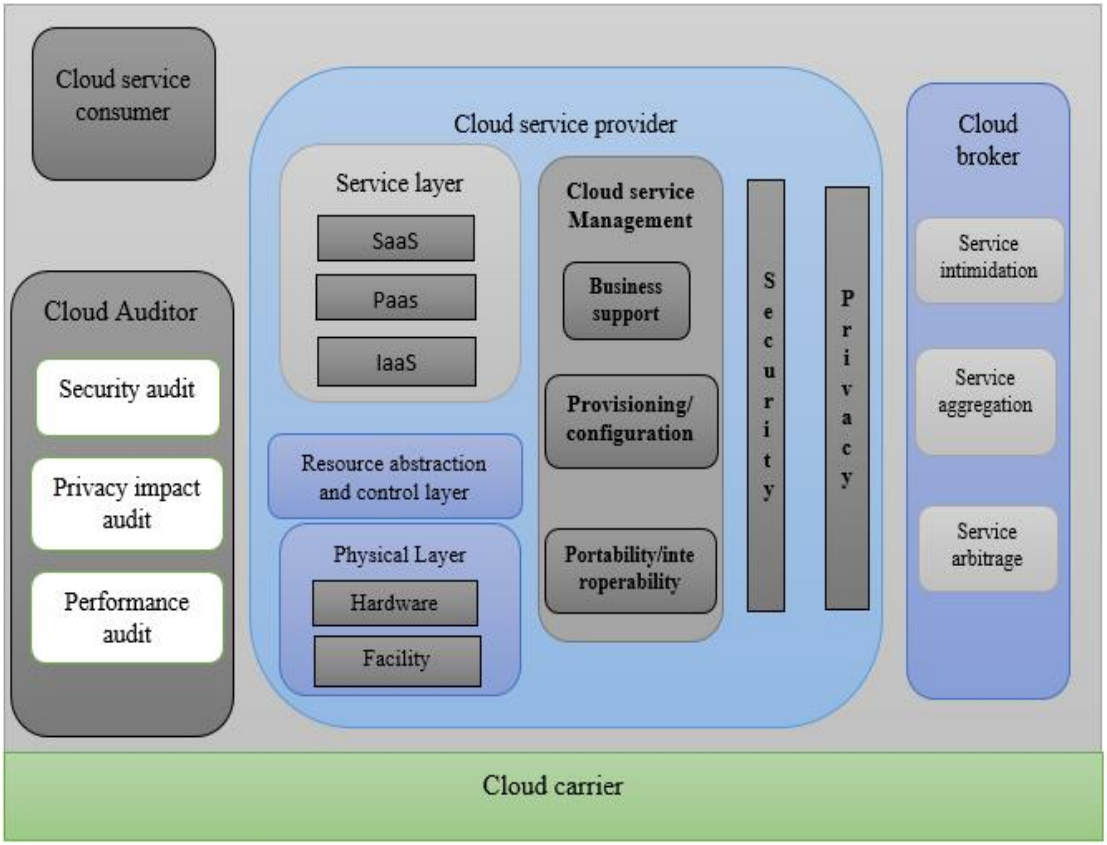

Figure 2. 1: NIST Cloud Computing Reference Model

2.5 Cloud computing models

2.5.1 Service model

Now a day, a wide variety of cloud services may be provided at a marketplace of IT utilities. Cloud computing service models are composed from many different components. This encapsulates cloud infrastructure and has an application programming interfaces and there are available across the network. The cloud services reflect any kind of IT capability that cloud service provider (CSP) offers to cloud service users (CSCs) [23]. As cloud computing provides services to users over the internet, it has three categories [24] such as IaaS, PaaS and SaaS. Figure 2.2 shows the overall cloud service models as well as service class, management tools and service content as described in [24].

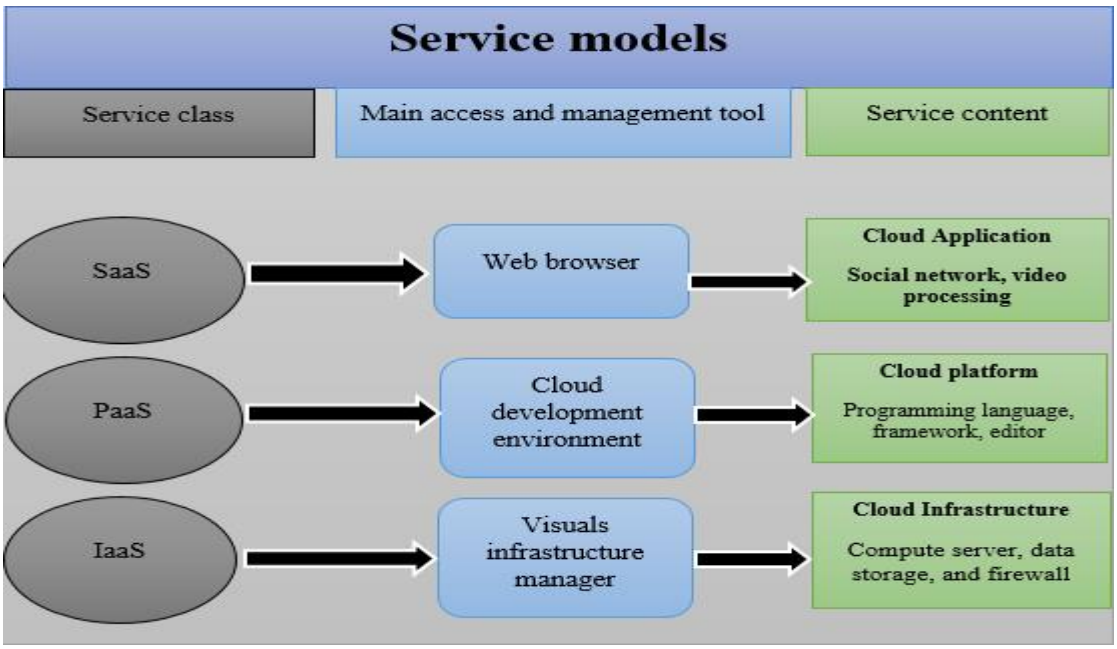

Figure 2. 2: Cloud service models

Infrastructure-as-a-Service (IaaS)

In the cloud technology, infrastructures are accessed remotely via IaaS. This model helps to access cloud storage and network space from remote servers. In IaaS, the user can deploy new user as a guest operating system (OS) and application. The user does not operate the entire cloud infrastructure, but manages the operating system and the storage. It can also deploy applications and various components of the network. EC2 and GoGrid are a few of the amazon IaaS providers. In IaaS, the developers use remote infrastructure to allow users to run whatever

an applications they want on their own choice of cloud hardware [25]**.** Figure2.3 indicates that the responsibility of cloud subscriber and cloud service provider with in IaaS [25].

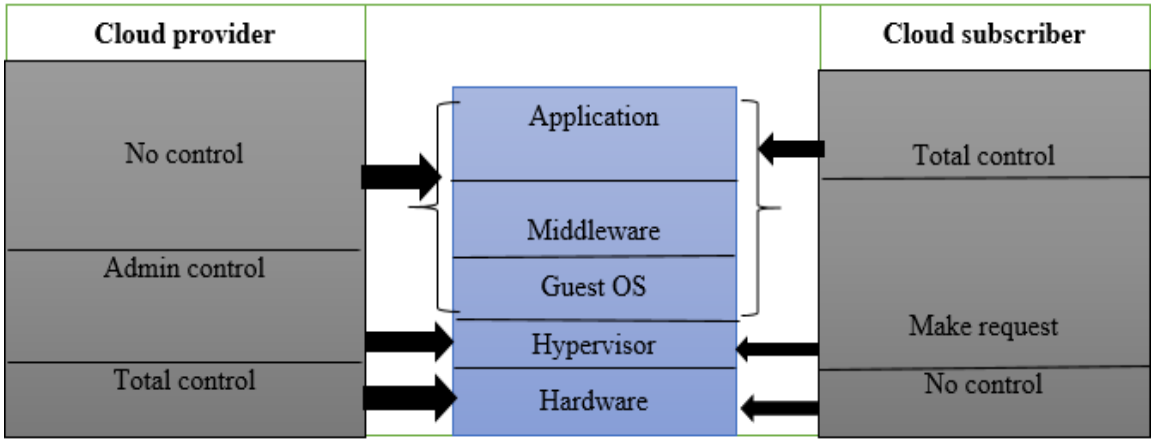

Figure 2. 3: IaaS provider and subscriber control responsibility

Platform-as-a-Service (PaaS)

PaaS model provides the customer with the ability to deploy customer built applications on the cloud platform based on programming languages and provider supported development tools. Under PaaS, the customer does not control the cloud infrastructure that underlies cloud. PaaS can provide platform to users to work on web applications or software. PaaS enables users to build their own cloud applications using software and language, which are unique to their suppliers [25]. In the PaaS model, the client is required to write an applications running in the particular environment of the provider. Figure 2.4 in [24] illustrates the role and responsibility of cloud user and provider in PaaS model.

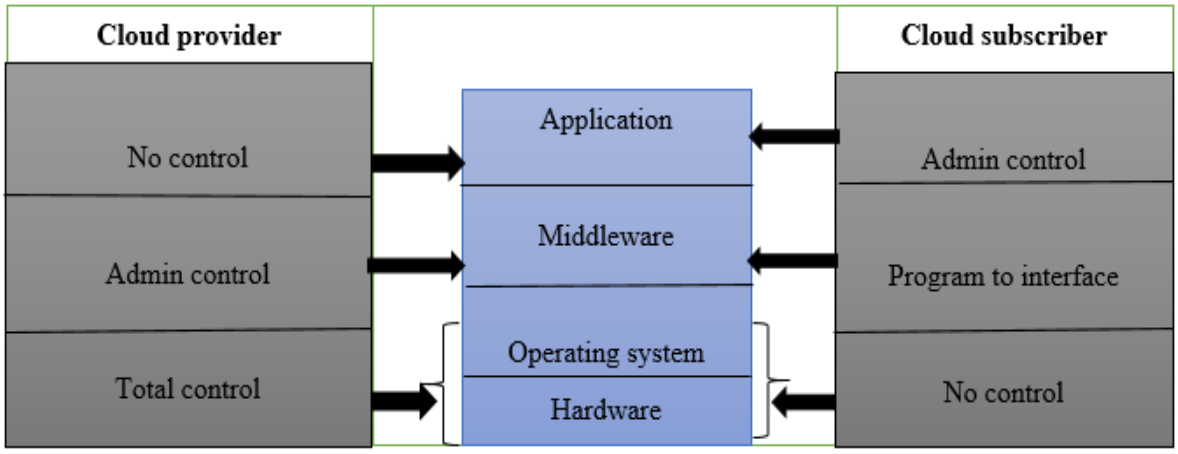

Figure 2. 4: PaaS provider and subscriber control responsibility

Software-as-a-Service (SaaS)

SaaS is the top layer of the cloud computing business model and it is browser initiated application services. Through SaaS, the company does not need to invest through servers or licensing tools. SaaS is often referred to as on-demand program and typically pay-per-use billing, pricing model for customers. It is the most common form of cloud service that small offices are able to use.

A service provider manages an application in SaaS and then customers access it through the World Wide Web. SaaS covers Microsoft Office 365, Salesforce, Google Apps and Google Docs [24].

Under SaaS, the cloud provider is authorized to control the level of the program. The user does not monitor or manage middleware, hardware or operating systems. As shown below, Figure 2.5 illustrates service provider and cloud user management roles in SaaS model [24].

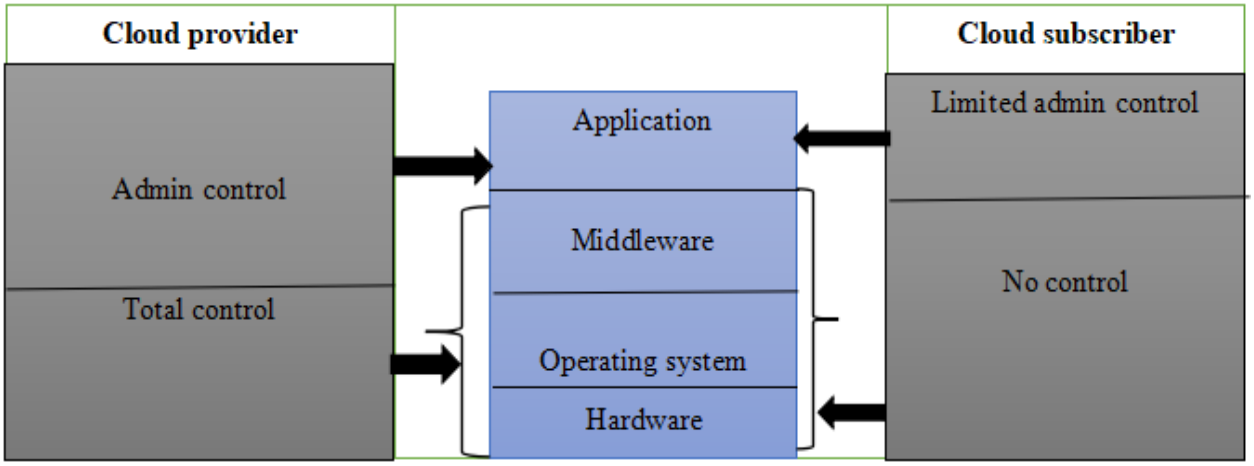

Figure 2. 5: SaaS provider and subscriber responsibilities

2.5.2 Deployment models

Cloud deployment [25] refers to where the cloud infrastructure is geographically located and managed. Selecting the appropriate type of deployment model is very important for an organization. For an entity or an organization, it is very important to choose the right type of cloud computing deployment model. Organizations must analyze their data precisely before deciding which type of model is chosen in order to avoid failure of implementation. There are four types of cloud deployment models. Those are private cloud, public cloud, hybrid cloud and community cloud [25].

i. Private Cloud

In the case of private cloud, the infrastructure of cloud is solely run for an entity and the service is not accessed by public group. In private cloud the resource is only available for a specific single company providing that company with better control and confidentiality. Private clouds provide greater security than public clouds [25]. Some characteristics of a private cloud [26] are listed below.

Improved security measures: It is one of the IT industry standards. Private cloud ensures security against unauthorized use, such as hacking by offering strong security tools.

Dedicated resources: Businesses centers have their own dedicated resources in private cloud, such as processor time and data buses that ensure ideal execution.

Better customization: The private cloud platform can be customizable because it can be designed to meet the commercial institution's ideal needs.

ii. Public Cloud

Public cloud provides data storage, applications and other services to its users. It is owned by the service provider based on the characteristics of pay-per-usage model. Public clouds are open to the public or organizations and it is managed by third party of cloud service provider. Public clouds have certain disadvantages like privacy and data protection. When users use public cloud, they do not know where the data is located. The characteristics of public clouds [26] are described below.

Flexible and elastic system: Public cloud provides the users with a scalable framework. This helps for consumers to exchange and store their information according to their requirements.

Freedom of self-service: Public cloud conjures up its customers in making a cloud all on their own exceptional taking of anybody's assistance. This is due to the preconfigured clouds that exist on the internet. The key factor of using public cloud is that privacy.

Pay-Per-Use: This unique feature enables companies to work in a coordinated manner to make the cloud usage more available.

Availability and reliability: The users have the opportunity to take any time for their work from anywhere in the globe.

iii. Hybrid Cloud

Hybrid cloud is made up of lots of public and private cloud shared between institutions with common priorities and needs. This can be handled internally either by an internal or external host, which is third party. Hybrid cloud infrastructure is a composition of two or more clouds, which may be separate entities and are collectively bound by the use of standardized or proprietary technology that allows portability of data and applications [27].

Hybrid cloud is supplied by one of two methods. The first is that the vendor has a private cloud and is partnering with a public cloud service provider. The second approach is that the public cloud provider collaborating with a company running on private cloud systems [27]. Hybrid cloud has its main characteristics [26], like that of the public and private cloud.

Optimal use: Hybrid cloud can expand server usage by scaling the open assets to keep the host running.

Availability: In addition to its expensiveness, accessibility on the corporate server is difficult, as it requires data strengthening, data redundancy and geographic scattering. In a hybrid cloud, if the institution's server is not available due to any defects, the total public cloud may theoretically scale up or fully overwhelm the operations.

Risk Transfer: Organizations can manage and run their server. The provider of the public cloud must ensure an intense of uptime for their service. Using the hybrid cloud, the danger of misestimating workload is relocated to the cloud dealer from the service operator.

iv. Community Cloud

The community cloud is overseen and used by different number of institutions that have the same core business, initiatives or shared needs infrastructures that include software and hardware. Community cloud is a hybrid form of private cloud. They are multi-tenant platforms that enable different organizations to work on a shared platform. Community cloud is used and operated by an organization group for some specific concerns [26]. The purpose of community cloud is allowing multiple customers to work on joint projects and applications that belong to the community, where it is necessary to have a centralized cloud infrastructure.

2.6 Virtualization in cloud computing

Virtualization is responsible for creating, migrating and canceling virtual machines in cloud technology. It plays an important role in cloud computing models. Virtualization is the

strategic aspect of cloud computing that enables the physical resources can use by multiple customers. It creates the virtual resource or device instance such as OS, servers, network resources and storage devices where the framework uses more than one execution environment to use the resources. Virtualization encompasses both software resources and hardware resources and makes the cloud computing more popular [21].

Virtualization technology makes handling the resources simple for the cloud computing world. In a single hosting environment, it can abstract and isolate the primary hardware and networking resources. Virtualization can also enhance cloud computing security by protecting both the integrity of cloud components.

In virtualization, on-demand virtual machines can be scale-down or scale-up. It enables sharing of resources, fast provisioning and isolation of workloads. The trends of virtualizations are consolidation of datacenters and thus reducing managing of cost. In addition to its advantages, it has some downsides because managing virtual resources is crucial and it is difficult to achieve high availability these resources [21].

The hypervisor is a key component of the virtualization process. Hypervisor is software, hardware, or firmware, which provides virtual partitioning capabilities that run directly on hardware. It is the virtual machine manager, which allows multiple OS to run on a system at a time providing resources to each OS without any interaction. All of the guest systems are managed by Hypervisor. Managing is difficult as the number of OS increases and this can lead to security issues. When a hacker gets control of the hypervisor the hacker can monitor the guest systems by knowing the system's behavior that causes damage to data processing [20]. Figure 2.6 shows the virtualization mechanism in cloud computing environment [21].

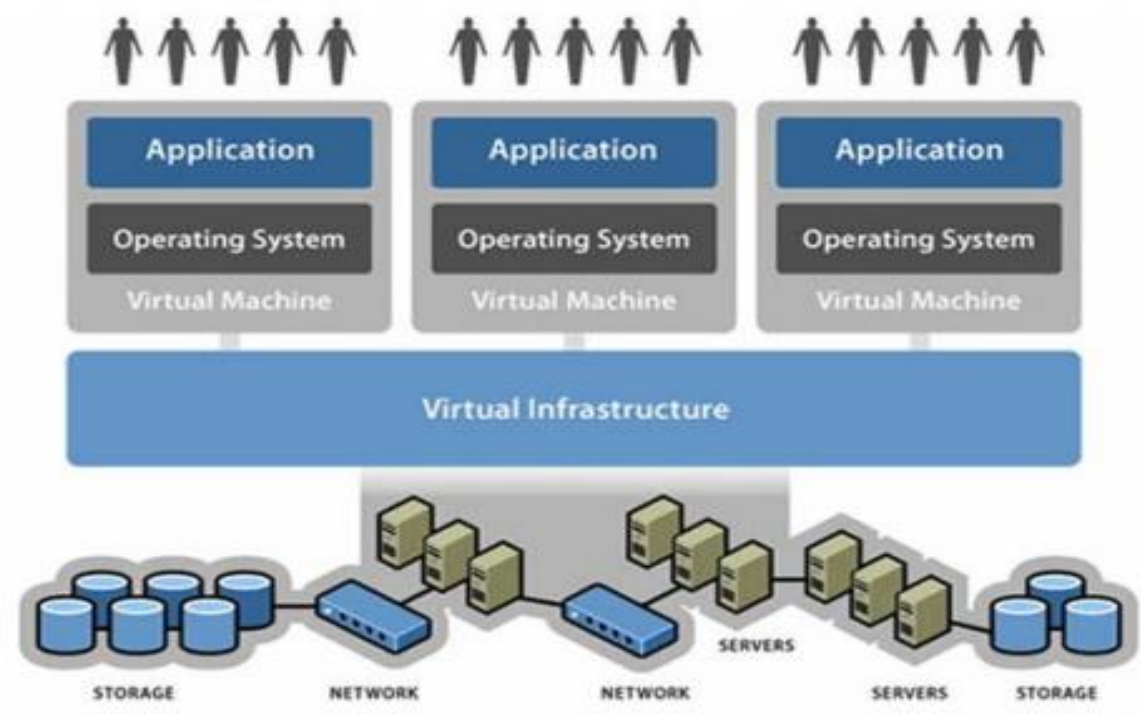

Figure 2. 6: Virtualization mechanism

2.6.1 Types of virtualization

There are different types of virtualization in cloud computing [28].

i. Server virtualization

Virtualization of servers is partitioning of a physical severs into smaller virtual servers to help maximize server resources. It makes one server appear as many servers. Each virtual server can run the same operating systems or different ones. A single physical server is virtualized to form multiple virtual servers to decrease server idle times.

ii. Storage virtualization

Storage virtualization is the pooling of physical storage from multiple network storage devices in to what appears to be a single storage that is managed from central storage. Many network storage facilities are available in storage virtualization as a single storage system, which is operated from the central console that can make more effective and simpler. Storage virtualization is abstracting, hiding or isolating the internal functions of a storage system from the host computers or general network resources, for the purpose of permitting application and network independent management of storage.

iii. Network virtualization

Network virtualization is the process of integrating the infrastructure and features of the hardware and software network into a unified virtual network. It enables network scalability optimization, data transfer speeds, security, flexibility, reliability and also automates some administrative activities related to the network. Network virtualization results for improved network efficiency and productivity.

iv. Desktop virtualization

This is the method of virtualizing desktop computers in the datacenters and these virtual desktop environments are used to communicate with a virtual machine on the network users in the same manner as a real machine is accessed and used. This is also referred to as device virtualization, which is used to separate a computing system from the actual machine.

v. Memory Virtualization:

Memory virtualization [28] is sharing the physical memory of the system and dynamically allocating it to virtual machines. This consist of a method to isolated memory from server to provide a networked, distributed or shared function. It improves the performance of the cloud system by giving larger memory capacity without addition to the main memory.

vi. Software virtualization

It provides the capability for a computer system to create and run one or more virtual environments. Software virtualization is used to facilitate whole computer systems to enable a guest OS to run and it used to allow remote access to the user screen from any location [17].

vii. Data virtualization

Data virtualization is a data management technique that allows an application to access and manipulate data without needing technical data information, such as how it is stored at source or where it is physically located.

2.7 Characteristics of virtualization

As virtualization is necessary in cloud environment it has the following characteristics [29].

Consolidation: Virtualization eliminates the need of dedicated single system to one application and multiple OS can run in the same server. Also, it helping the developers isolates different applications in their respective virtual partitions.

Easier and flexible for development: Application developers may able to run their applications and programs in heterogeneous OS environment on the same virtual machine. Isolating of different applications in their respective virtual partitions also helps the developers.

Stability and security: Host OS can host different types of multiple guest OS containing multiple applications in a virtualized atmosphere. Each virtual machine is isolated from each other and they are not interfering in to the other's work, which in turn helps the security and stability aspect.

Migration and cloning: Virtual machines can be transferred to balance the workload from one platform to the next. Due to migration, users can access updated hardware and recover from hardware failure. Clone is a copy of an existing virtual machine. Cloned virtual machines are easy to deploy at both local and remote locations.

2.8 Virtual machine migration

One of the most significant aspects of virtual machine technology is migration of the virtual machine [30]. It involves transferring a virtual machine from one physical host to another, to enabling the load balancing, resource management and reducing energy usage. It enables system administrators to switch an OS instance to another physical node interrupting any moving OS hosting services. Virtual machine migration often needed to migrate virtual machines due to any system fault in the event of unscheduled server downtime. It can also be used for disasters recovery. There are various benefits of virtual machine migration such as load balancing, server consolidation, hotspot and cold spot migration [30].

Load balancing: This decreases the inequality of resource usage levels across all the physical machines in the cluster. This prevents some machines from getting over-loaded in the presence of lightly loaded machines with sufficient spare capacity. Migration of virtual machines from over-loaded physical machines to under-loaded physical machines can balance the overall system load.

In load balancing method, the load balancer is responsible for balancing the load on physical machines in the datacenter, which resides in the central controller. In addition to this, it is responsible for making detection of over-loaded physical machines, detection of under loaded

physical machines, selection of best virtual machines to migrate and selection of best hosts for migrating virtual machines.

Server consolidation: server consolidation algorithms are needed to minimize server sprawl in datacentres. These algorithms are virtual machine packing heuristics, which attempt to pack as many virtual machines as possible on a physical machine. So that, the resource use can be improved and machines that are unused or underused can be turned off. Consolidation would result in lower power and thus reducing overall operational costs for datacenter administrators.

Hotspot and cold spot migration: Hotspots and cold spot detection are always based on thresholds set by the owner of the datacenter or the clients specified by SLAs. Physical machines having resource usage values beyond the upper threshold are said to be hotspots and whose usage values below the lower threshold are said to be coldspot. The former implies overuse and the latter implies under-use, which can be applied across any dimension of resources.

2.8.1 Types of virtual machine migration

There are two types of virtual machine migration [30].

i. Non-live migration

This type of migration is also called cold migration. It is the method of migrating the guest operating system and the applications shutting down the OS and then restarting the operation of a virtual machine at the target host. Non-live virtual machine migration [31] is not suitable for interactive web applications, because it can causes performance degradation for interactive applications and also it cannot meet the strict SLA requirements any more due to its long service downtime.

ii. Live migration

This type of virtual machine migration can allow virtual machines to be moved from one physical host to another, while continuing executing even after migration, without any loss of connection with the user. It requires moving all the state information of the virtual machines being migrated (memory status, network status and storage status) from one physical server to another within the same datacenter or over remote datacenter.

In cloud datacenters, virtual machine live migration is usually done for the purposes like load balance, power management, hardware maintenance and system up gradation. The aim is also

used to distribute load across the physical servers to enhance virtual machine scalability, reliability and availability. There are two types of live migration techniques called pre copy and post copy.

Pre copy approach: It used for migrating virtual machine memory state. In this migration type, first transfers the memory state and then the central processing unit (CPU) state. This type of approach has two phases. The first phase is push warm up phase and the second one is stop and push copy phase. The hypervisor copies the entire memory page from source to destination in the warm-up push phase, while the virtual machine still runs at source. If some memory pages are altered, dirty page will be created through the memory copy process and it will be recopied. In stop and copy phase, the source virtual machine is stopped, pages are copied across to the destination virtual machine, then the new virtual machine is started.

Post-copy approach: Where the virtual machine is stopped immediately, a minimal processor status is copied and the virtual machine resumed at the destination after which any memory page provided by the running application is pulled from the source. Post copying will minimize the overall migration period, as each memory page is only transferred once, it can minimizing downtime.

2.9 Scheduling in cloud computing

Cloud computing is the use of computing resources as internet service. Scheduling is an important activity that can assign tasks to resources in cloud. Cloud computing scheduling refers to the method of transferring a set of jobs to a set of virtual machines or allocating virtual machines to operate on the resources available to satisfy the needs of users. Scheduling is one of the tasks performed to get the most extreme benefit inorder to increase the efficiency of accessing cloud resources. The primary aim of cloud-based scheduling algorithms is to use resources efficiently while handling the load between resources in order to get the least execution time [32]. Therefore, when users want to develop scheduling, there are several key factors, such as equal allocation of resources, maximizes the usage of resources and decreases energy consumption.

Throughput, turnaround time, waiting time, response time, utilization of CPU, computational complexity (job length and processing power) and computing cost (processor cost) are the

major key factors of scheduling [33]. One or more techniques should be used to develop each scheduling technique.

Resources in cloud computing are scheduled at two levels [25]. The first one is at virtual machine level and the second one is at host level. At the virtual machine level, tasks are assigned for execution to the allocated virtual machines using a job scheduler. It is called task scheduling. At the host level, a virtual machine scheduler is used to allocate the virtual machines into physical hardware. This type of scheduling is usually called virtual machine scheduling. Task scheduling concentrations are mapping of tasks to appropriate virtual machines efficiently. Whereas, virtual machine scheduling is allocating of virtual machines to run on the appropriate physical machines to insure the implementation of tasks. This can improve the utilization of resources as well as managing the load of the systems**.** Virtual machine scheduling is important to ensure the QoS and SLA agreed by the cloud service providers and customers.

There are two policies for scheduling in cloud computing [34] called space-shared scheduling and time-shared scheduling. Both policies can be used to virtual machine scheduling and task scheduling. In space-shared scheduling policy, only one virtual machine or task is allowed to be executed at a given instance of a time on host or on virtual machine. In time-shared scheduling policy, it allows multiple virtual machines or tasks run simultaneously within a host or virtual machine.

The process of scheduling in the cloud can be done in three stages [35].

Resource discovering and filtering: Datacenter broker know the current status of all the resources that are available in the cloud and also the remaining resources that may be available. Actually, these resources are generally the virtual machines. It regularly collects the status of each resource attached to the cloud [36].

Resource selection: Based on information obtained from the resource's status regarding current queued jobs and information on the status of cloud resources, the cloud scheduler makes decisions regarding the creation or deletion of specific cloud nodes (virtual machines) in order to best suit for run jobs.

Job submission: In this stage, finally the job is submitted to best available selected resource [36].

2.9.1 Scheduling Type

Based on the description of scheduling in [37] are divided in to various form.

i. Centralized and decentralized scheduling

A scheduling is called centralized, the decisions are made in a central node. Centralized scheduling insures efficiency and ease of monitoring resources. However, it has lack of scalability and fault tolerance. Decentralized scheduling is also called distributed scheduling, which is more applied in real cloud environment and it has lack of efficiency [37].

ii. Static and dynamic scheduling

Static scheduling is done before the system begins operating. In static scheduling, all timing information about tasks is available before any action performed. So, the execution schedule of each task is computed before executing any task. This type of schedule is effective for applications that have fixed demands. Moreover, in static scheduling, the customer makes an agreement with the cloud provider for services and the cloud provider prepares the required resources before the start of the required service [38]. But, in dynamic scheduling, the timing information about the tasks is not known. Therefore, the execution of the tasks may change as per the user demand. Dynamic scheduling incurs runtime overhead compared to static scheduling. Dynamic scheduling does not allow the cloud provider to plan before usage and it assigns and removes resources as per needed [39].

iii. Preemptive and non-preemptive scheduling

Preemptive scheduling permits interrupting of each task during the execution and transferring of the task to another resource. This type of scheduling is mandatory if constraints need to be imposed such as priority, deadline and cost. In non-preemptive scheduling, the virtual machine cannot be taken away until the task running on it completes. It does not allow the task to be interrupted while it is executing [39].

iv. Immediate mode and batch mode scheduling

Immediate mode scheduling is also called online mode scheduling, which tasks are scheduled to resources immediately without any delay. Tasks are scheduled only once and cannot be changed. On the contrary, the batch mode collects tasks into a set and are examined for mapping at prescheduled times. It is called offline mode scheduling [40].

v. Heuristic and metaheuristic scheduling

Heuristic scheduling techniques are problem dependent, which can solve particular problems. Metaheuristic techniques are high-level problem independent techniques that provide master strategies to solve general problems and can be applied to a wide range of problems. Both scheduling techniques do not guarantee that they will find the optimal solution, but they provide a good solution when compared with the time spent on computation [41].

2.9.2 Job scheduling

Job scheduling states that the dispatching of computing tasks to resource pooling between various resources users depend on certain rules. There is no uniform standard for job scheduling in clod environment. Job scheduling is performed using job schedulers that are programs, which enable scheduling [6]. It is the process of how jobs should be executed in the environment in terms of actual mapping of resources and time when the jobs are executed. Job scheduling is a critical component of the cloud resource management. In cloud computing, job execution involves specific resources that are available to them by meeting certain constraints such as best performance, minimum execution time, shortest response time, fault tolerance and expected service quality. Under cloud computing, scheduling can be viewed into two ways. From the user's point of view, the scheduling algorithm should reduce the time of processing and the user's budget. On the other hand, from the cloud provider's perspective, planning of algorithms should improve resource of utilization, reduce the cost of maintenance and energy consumption [39]. Because the main aim of job scheduling is distributing the system load, improve system performance, proper utilization of available resource and reduce the cost as well as execution time. It has biggest problem that sometimes multiple requests are come at the same time. This can cause the system cannot work properly [6]. Due to that many problems can occurred like system delay and make job scheduling is a combinational problem. It cannot be considered as linear programming and it is impossible to find a global optimal solution by using a simple algorithm or rule.

2.9.3 Scheduling problems

Scheduling problem [42] is the problem of matching elements from different sets, which is formally expressed as a triple (E, S, O), where ***E*** is the set of examples, each of which is an instance of problem, ***S*** is the set of feasible solutions and ***O*** is the object of the problem. Scheduling problem can be classified into two categories depending on object ***O***. These are

optimization problem and decision problem. Finding the best solution among all the feasible solutions in set $\boldsymbol{S}$ requires an optimization problem. In the decision problem, the answer of output is yes or no. Unlike optimization, the decision problem's objective is relatively simple. For a specified feasible solution $\boldsymbol{s} \in \boldsymbol{S}$, the problem needs positive or negative answer whether the object $\boldsymbol{O}$ is achieved. Obviously, the optimization problem is harder than a decision problem.

2.10 Schematic methods for scheduling problem

Scheduling problems belong to a wide variety of combinational optimization problems aimed at finding an optimal match solutions from a finite set of objects, such that the list of feasible solutions is typically discreet rather than continuous. An easy problem refers to one with a small number of examples, so it can be simply worked out by polynomial algorithms or enumerations.

On the contrary a problem is in class of non-deterministic polynomial (NP) complete[1], if its purpose is making a decision. In addition, a problem is in class of non-deterministic polynomial (NP) hard, if its purpose is optimization. Because an optimization problem is not easier than a decision problem, we only list schematic methods for NP-hard problems[2]. As shown in Figure 2.7, enumeration, heuristic and approximation are three possible solutions of NP-hard problems [38].

[1]If a polynomial time algorithm exists for any of these problems, all problems in NP would be polynomial time solvable. These problems are called NP-complete.
[2]A problem is NP-hard if all problems in NP are polynomial time reducible to it, even though it may not be in NP itself.

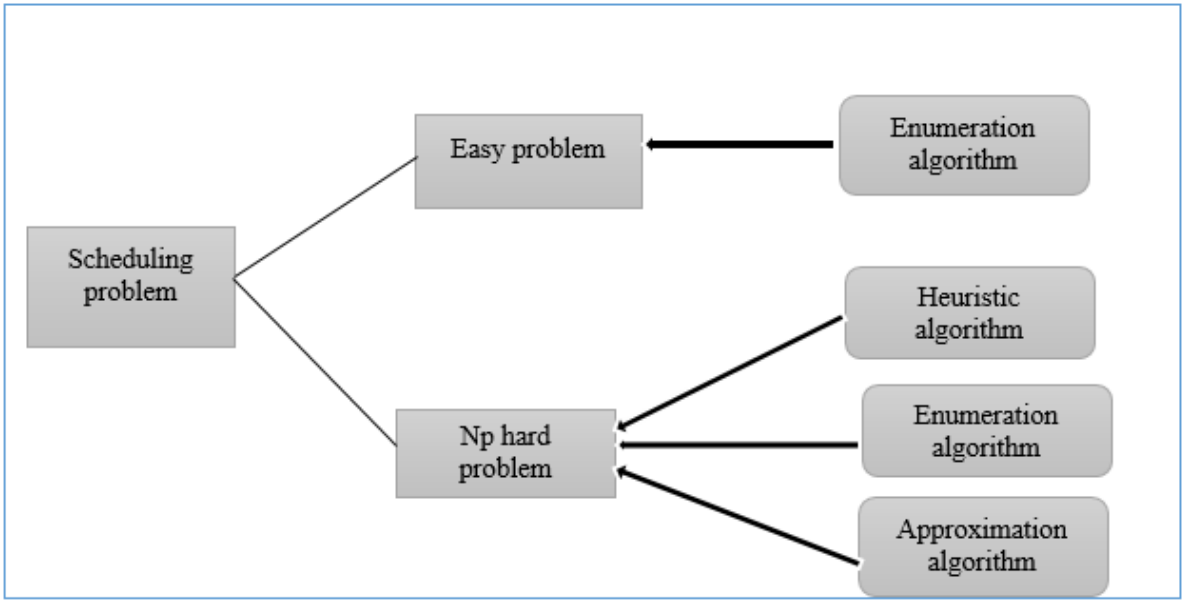

Figure 2. 7: Schematic view of scheduling

As shown in Figure 2.7, easy problem can be solved by enumeration algorithm. Whereas, NP hard problem can be solved by the method of heuristic, enumeration and approximation algorithm.

Easy problem: - It is solved by enumeration method. In enumeration method, the optimal solution can be chosen for an optimization problem if all the possible solutions are enumerated and compared one by one [38]. Exact enumerative algorithms have the exponential time complexity in the worst case. However, for some NP-hard problems in other sense, when the number of instance is relatively small, it can be solved by a pseudo polynomial algorithm, and the time complexity of which is bounded by a polynomial expression of the input size and the maximum number of the problem.

NP hard problem: - It is solved by method of heuristic, enumeration and approximation algorithm. When number of instances is large, exhaustive enumeration is not feasible for scheduling problems. In that case heuristic is a suboptimal algorithm to find reasonably good solutions reasonably fast. Heuristic is a suboptimal algorithm to find reasonable good solutions quickly. It iteratively improves a candidate solution with regard to a given measure of quality, but does not guarantee the best solution. Approximation algorithms are used to find approximate solutions to optimized solution [38].

2.11 Service Level Agreement

The services of c1oud computing can be depending on pay based usage, which is based on the agreement between service provider of the cloud and service user of the cloud. SLA is a major issue in cloud computing that defines important parameters and specifications. SLA [43] is a legal contract between participants, which a contract is a legally binding agreement between groups of parties. Contracts are subject to specific legal interpretations. SLA exists in many forms with different metrics and measurement methods. For example, it can measure at per customer level or per query level. There are so many parameters of SLA [43] such as response time, waiting time, availability, security, location of data, disaster recovery expectations, period of service, pricing, customer level satisfaction, bandwidth, storage, reliability, migration, load balancing, deadline, resource utilization, throughput, CPU and RAM capacity, cost, storage and so on. Also threshold value can be taken as SLA, which is boundary value measured over the operations result [44].

However, the service varies from one CSP to another [13]. SLA is used in cloud provider and cloud service customer in order to ensure the following points.

- ✓ QoS requirements are meeting in SLA and if any party violates the SLA terms, the defaulter has to pay penalty according to the clauses, which are define in SLA.
- ✓ SLA is applied in scheduling in order to indicate the profits that service provider may obtain if the service is deliver at certain optimal levels.
- ✓ SLA can insure the agreement negotiated between service users and providers, which defines the metrics of expected QoS and penalties during service delivery.

A schedule, which is based on SLA, can schedule the task to the appropriate machine depend on the agreement set between cloud parties and periodically checking for SLA violation [45]. SLA in cloud environment mostly includes:

- ✓ Responsibilities of both providers and customers.
- ✓ List of services and their definition, which the provider provides to the customer.
- ✓ Agreement provided by the provider regarding functional and non-functional requirements.
- ✓ Legal context, which the provider and customers have negotiated.

S.B.Dash *et al*., [46] in Figure 2.8 shows clear representation of SLA between cloud provider and cloud service user as shown below.

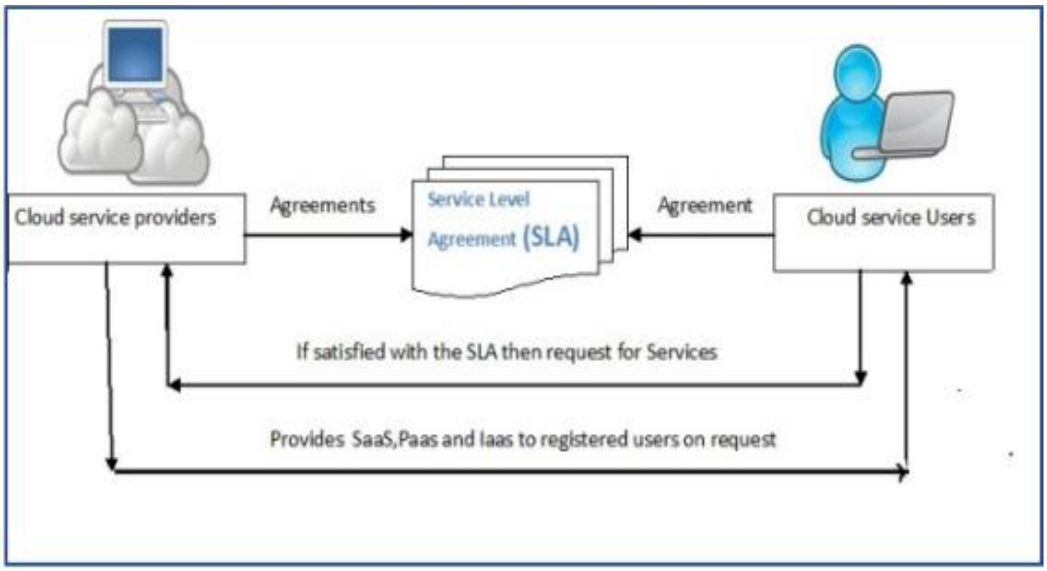

Figure 2. 8: Service level agreement of cloud

2.11.1 Life cycle of SLA

SLA has six phases based on the report of European commission on national projects about SLAs cloud computing [45].

- ✓ **Service use (first phase)**: It handles the information that affects the cloud service usage by the service customer.
- ✓ **Service modeling (second phase)**: It deals with the modeling of the service, relationship and dependencies within the service components and information regarding the service provision.
- ✓ **SLA template definition (third phase)**: This is the third phase, which deals that SLA templates are created and other related information is captured.
- ✓ **SLA management (fourth phase)**: This phase deals with the management of SLA covering various aspects such as SLA definition, SLA modeling, SLA negotiation (including SLA re-negotiation after the service provisioning in cloud), SLA monitoring, SLA evolution and SLA violation and trustworthiness.
- ✓ **SLA enforcement (fifth phase):** The purpose of this phase is to enforce the SLA.
- ✓ **SLA conclusion (sixth phase):** It handles the termination of the SLA, which can happen for various reasons such as SLA violation or expiry of the service period. Figure 2.9 in [45] shows that the life cycle of service level agreement in cloud computing.

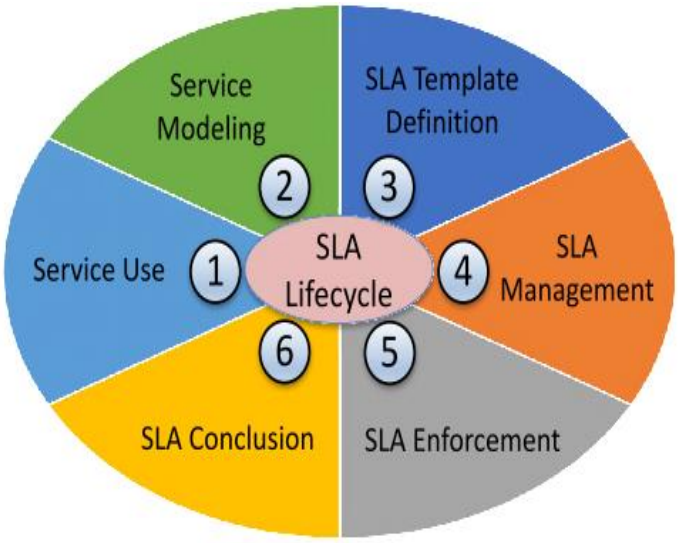

Figure 2. 9: Life cycle of service level agreement

2.11.2 Importance of SLA

SLA [21] has many importance within the technology of cloud computing for cloud parties.

- ✓ SLA enables to the customer that can get the information about the service providers.
- ✓ SLA describes the complete information about the services and the type of services (SaaS, PaaS and IaaS) that are provided to a particular customer.
- ✓ SLA describes the purpose and objectives based on business level policies, which includes the part of the service provider and the customer.
- ✓ The customers will be able to identify the QoS and management strategies of the agreement.
- ✓ SLA is used to monitor the QoS parameters from the service point of view.
- ✓ The customer can get the idea about the requirements for the management of the service in case of poor performance.

2.11.3 Features of SLA

In cloud computing SLA has some features. According to [21] SLA in cloud computing environment has three main features.

i. **Enhanced the satisfaction level of customer**

A healthy defined SLA raises the customer satisfaction level, as it helps the provider to focus on the requirements of the customer and warrants that the effort is placed in the right direction.

ii. **Amended service quality**

QoS requirements [47] are commonly identified in SLAs. SLA consists of a contractual agreement between service providers and customers that fact the nature, worth and objective of the service to be provided. SLA may also identify the percentage of violations that can be accepted within a predefined time interval, before the service provider experiences penalty.

iii. **Enhanced relationship between two parties**

A healthy SLA specifies the compensation and penalty policies of the service. The customer can observe the services according to service level objectives (SLO) [3] identified in the SLA.

2.11.4 Classification of SLA

SLA in the cloud environment has three kinds [21].

i. **Service based SLA**

It is the first type of service level agreement. Service based SLA refers the contract made between the service provider and the customer based on the required services to be delivered. In this type of SLA, the service level of the customer query service will be the same for all customers that will be using this service.

ii. **Customer based SLA**

It is the second type of SLA that includes all the required services that are needed to a company and it is appropriate to an individual customer group. Like, the SLA set between the service provider and the finance department of a big company for the required services such as finance system, payroll system, billing system, procurement or purchase system.

iii. **Multi-level SLA**

This type of SLA focuses on the organization of the customer. When defining the multi-level service agreement structure, all the services and their interrelationships with subordinate services are used.

---

[3] An SLO (service level objective) is an agreement within an SLA about a specific metric like uptime or response time.

In general, designing scheduling for cloud environment using SLA is more desirable to achieve high level of user satisfaction by considering different QoS parameters as an agreement between customer and providers.

2.12 Summery

In this chapter, we have reviewed literatures of cloud computing environment with its concepts. Additionally, the major characteristics, architecture, various models of cloud computing, virtualization and migration of virtual machine is discussed. Besides this, the various types of scheduling in cloud computing are also presented. The detail description of SLA with in cloud computing and the role of SLA also described. As a result, from this chapter, we have a great understanding to facilitate the solution of the domain specific area of our research work.

Chapter Three: Related Work

This Chapter deals with an overview of research works about some scheduling methods, which is done based on service level agreement. Each scheduling algorithm which are based on SLA are briefly described with their various contributions, parameters used. Finally, the common gaps of the reviewed works as well as the way the newly proposed algorithm fills those gaps is described briefly.

3.1 SLA-aware resource scheduling for cloud storage

SLAs are one of the major considerations for every buyer of cloud computing service [11]. In order to increase the market share in the highly competitive cloud market, cloud providers must offer higher service and differentiated performance SLAs in terms of input/output (I/O) throughput, availability, capacity etc. SLA aware resource scheduling for cloud storage [11] addressed the problem of design a scheduling policy to effectively utilize storage resources and manage the performance in a cloud infrastructure platform. As a result, the cloud storage services can be offered with guaranteed capacity and I/O throughput.

Yao, Z. *etal.* [11] designed and implemented SLA aware scheduling policies based on the distributed OpenStack[4] scheduling model. The paper in [11] had designed a scheduling policy to effectively utilize storage resources and manage the performance in a cloud infrastructure platform on cinder services, which is the block storage of service block in open stack. The cinder services select the best candidate storage based on filtering and weighting (cost).

[4]Open stack is a free and open source computer program stage for cloud computing, for the most part sent as IaaS, whereby virtual servers and other assets are made accessible to the client.

After the filtering stage, the request can be served by all hosts on the list. The scheduler calculates the cost, which is the weight of each host by comparing the hosts' characteristics and the request characteristics. Then hosts on the list are sorted according to their weight. The host, which has the least weight, could be chosen as the best candidate.

As a result, cloud storage services can be offered with guaranteed capacity and I/O throughput. The beneficiaries of the approach in [11] were that the cloud storage providers would be able to provide higher I/O throughput performance of SLA to customers with less storage. The author of the scheduling in [11] takes storage is the only SLA parameters.

3.2 SLA-based scheduling of applications for geographically secluded-clouds

The geographic distribution [12] of cloud services is one of the several key factors that affect the net benefit of cloud computing. Cloud service providers have several datacenters at different geographical locations over the internet in order to optimally serve customers need. With the advances in technology, resource control is a significant challenge for geographically distributed clouds. Users who are geographically close to the server get enhanced service due to low latency. Depending on this concept, the paper in [12] proposed job scheduling algorithms in a distributed datacenter network, where the algorithms assign user's workloads to virtual machines hosted to datacenter close to the users to reduce response time.

The aspect of this proposed algorithm was, the use of delay to evaluate if a virtual machine provides a low or a high delay, it is required that the location of the end user producing the tasks should be known. When tasks are in the submission queue, the algorithm first assigns a priority value to each cloudlet. The value is based on the SLA framework considering response time. Tasks required short response time are sent to the virtual machines that are close to the service with low delay path. Here the paper [12] had listed the virtual machines as low delay virtual machines. Similarly, virtual machines with lower priority were listed as higher delay virtual machine that could provide higher response time. The algorithm was further enhanced by checking the current CPU usage of the virtual machines before submission. There were two proposed algorithms, which are based on time-shared approach. The one algorithm was called priority round robin (PRR) algorithm that provides priority to the task for short response time to finish quickly. It sends the task to a low delay path over distributed clouds. The second

one was called SLA priority round robin (SPRR) algorithm, which not only provides priority to the task requiring short response time but also enhances the QoS to ensure the SLA is adequately drafted. The new algorithms also show that sending the cloudlets to the virtual machines close to the user utilizes more of the CPU. The algorithm in [12] used response time as SLA objective and give priority for tasks which needs shortest response time.

3.3 SLA-based task scheduling algorithms for heterogeneous multi-cloud environment

The collaboration of the CSPs in the heterogeneous multi-cloud environment is strongly challenging. In multi-cloud environment, the CSPs may have various SLAs in providing their services. As a result, it is very difficult for the customer to select the CSP. One solution of this problem is that the CSPs can also collaborate with each other to provide services under a common SLA. It is difficult to scheduling all the tasks to the clouds with respect to their SLA. The paper [13] proposed SLA based task scheduling algorithms for heterogeneous multi-cloud environment for addressing the problem occurred in assigning tasks in heterogeneous multi-cloud environment. The problem was how to schedule all the tasks to the clouds with respect to their SLA. The problem also considers for balance between the overall processing time or makespan and the gain or penalty of cost for a given set of $\boldsymbol{l}$ number of independent tasks and a set of $\boldsymbol{m}$ number of clouds along with their execution time, agreement (gain) and penalty cost of the tasks on different clouds.

For achieving such problem the paper [13] was used three types of service levels to represent SLA called minimization of execution cost, execution time and both where execution cost is represented by means of gain cost and penalty of cost. In order to achieve such kind of problem, the proposed method in [13] had used two scheduling algorithms called service level agreement-min-min (SLA-Min-Min) and service level agreement minimum completion time (SLA-MCT). SLA-MCT is a single-phase scheduling or online scheduling, whereas SLA-Min-Min is two-phase scheduling (off-line scheduling). In the single-phase scheduling, a task had assigned to a cloud as soon as it arrives in the system. In the two-phase scheduling, the tasks are not assigned to the clouds as per their arrival in the system; instead, they are collected in a batch that is assigned at predetermined times. The proposed algorithms have enabled the customers to choose one of the SLA levels that aim to minimize the execution cost, execution time, or (both). The second contribution of the scheduling algorithm also facilitates the

customers to select the weight values for execution time and execution cost parameters. As a result, the customers can determine which parameter is more important. The result of SLA-based task scheduling algorithms for heterogeneous multi-cloud environment use performance metrics like makespan, average cloud utilization of the services.

3.4 SLA-based fault tolerant workload scheduling in cloud computing environment

In terms of application reliability, efficiency and fault tolerance, different users have various requirements. The authors in [14] covers the issue of fault tolerance, which is one of the most crucial issues which are faced by the cloud users and cloud service providers. A consistent degree of fault tolerance as well as overhead are the result of static fault tolerance policies. Static fault tolerance policies is unable to support flexible fault tolerance as per the user requirements. SLA-based fault tolerant workload scheduling solved the problem of rigid and static fault tolerance. Gill *et al*., [14] was proposed a method to implement adaptive fault tolerance job scheduling by considering customer requirements. The method used in the paper in [14] should be able to switch between the various faults tolerance solutions as per the user classification. The cloud users have been classified into sub classes as per the fault tolerance requirements. The proposed adaptive method can optimize the execution cost, turnaround time.

3.5 SLA-based Virtual Machine Scheduler in Cloud Environment

SLA-based virtual machine scheduler [15] in cloud environment was implemented due to lack of SLA in traditional Min Min algorithm. Because, Min-Min algorithm fails to utilize the resources efficiently which lead to a several problems like load imbalance, SLA and so on. The proposed task scheduling in [15] strategy aims at minimizing the total completion time as well as reducing the SLA problems for all tasks. The authors considers the process size, time and service type requirement of each task to realize the optimization for cloud computing environment. The algorithm [15] was have four performance metrics, namely makespan, average cloud utilization, gain, and penalty cost of the services in virtual machine. The paper focused on the efficient tasks scheduling considering the total completion time of tasks and resources utilization in cloud environment.

3.6 Service-level agreement-aware scheduling and load balancing of tasks in cloud

Kaippilly et al., [16] proposed a quality-assured SLA-aware load balancing and scheduling algorithm for the cloud environment. The algorithm in [16] migrates tasks from virtual machines from overloaded hosts and submits it to the virtual machines in the under loaded hosts having the highest capacity. A scheduling in [16] considered virtual machine bandwidth, virtual machine million instructions per second (MIPS) rate, and RAM capacity as parameters of SLAs. It considers SLA in both the initial scheduling stage and in the load balancing stage and it looks into different objectives to achieve minimum makespan, minimum degree of load imbalance and the minimum number of SLA violations. The proposed model in [16] uses a past usage pattern for predicting the resource requirement for optimal load balancing to reduce violations in SLAs.

3.7 Summary

As discussed above, this chapter covers the review of different types of scheduling, which considers various parameters as a key points of SLA.

As a result, SLA aware resource scheduling for cloud storage [11] is addressed the problem of design a scheduling policy to effectively utilize storage resources and manage the performance in a cloud infrastructure platform, so that cloud storage services can be offered with guaranteed capacity and I/O throughput. This scheduling policy is effectively utilized storage resources and manages the performance in a cloud infrastructure platform. The authors in [11] takes storage as SLA objective.

Biswas *et al*., [12] worked on the challenge of resource control for geographically distributed clouds. The paper in [12] was develop geographic distribution based scheduling algorithm worked on distributed datacenter networks, by assign users' workloads to virtual machines hosted to datacenter close to the users in order to reduce response time. SLA-Based scheduling of applications for geographically Secluded Clouds [12] used response time as SLA objective and give priority for tasks which needs shortest response time.

Panda *et al.,* [13] proposed SLA based task scheduling algorithms for heterogeneous multi-cloud environment. This proposed work solved the problem faced when customers to select the CSP. The main issue was enabling the customers to choose one of the SLA levels that aim to minimize the cost, execution time, or both cost and execution time. The paper proposed

two types of algorithm called SLA-Min-Min and SLA-MCT. The proposed algorithm can balance between the overall processing time or makespan. The paper in [13] was used three types of service levels to represent SLA called minimization of execution cost, execution time and both, where execution cost is represented by means of gain cost and penalty cost.

SLA based fault tolerant workload scheduling in cloud computing environment [14] aims to conduct the scheduling algorithm based on fault tolerance. The method used in this paper was aimed to implement adaptive fault tolerance job scheduling by considering customer requirements and taking fault tolerance as parameter of SLA.

SLA-based virtual machine scheduler [15] in cloud environment was implemented due to lack of SLA in traditional Min Min algorithm and it focused on the efficient tasks scheduling considering the total completion time of tasks and resources utilization in cloud environment. It used makespan, average cloud utilization, gain, and penalty cost of the services in virtual machine as performance metrics.

Kaippilly et al., [16] proposed a quality-assured SLA-aware load balancing and scheduling algorithm for the cloud environment. It considered virtual machine bandwidth, virtual machine million instructions per second (MIPS) rate, and RAM capacity as parameters for SLAs. Throughout of the review of the scheduling based on SLA, it achieves various parts of problems from different scheduling perspectives.

However, those scheduling are not including priority by computational complexity (job length) which is one of the major factors of scheduling [33] and also virtual machine migration together, as SLA constraint. Job length is the indication of task required to execute the instruction length [48] and it is the product of execution time and processing capacity of virtual machine. Execution time is the exact time taken to execute the given job and it depends on job length and processing capacity of the virtual machine.

The other consideration in this thesis is that virtual machine (Vm) migration which is used to optimize resource access for users in cloud environment. Vm migration can decrease the imbalance of resource use levels of the overall physical machines.

Therefore, this research proposes scheduling on cloud, which can schedule tasks based on the priority of SLA using job length and considers the virtual machine migration based on levels of host utilization for the purpose of load balancing.

Chapter Four: SLA-Based Job Scheduling Model for Cloud Computing Environment

This chapter discusses the architecture and the design consideration of SLA based job scheduling for cloud computing. There are two major parts called design consideration and system architecture. The design consideration used to determine in what way of the system can have better performance and the architecture of the system discusses the various components of the proposed scheduling model.

4.1 Design consideration

Through the design of the system, effective resource utilization and user satisfaction level would be achieved.

Effective resource utilization

By effective resource utilization of cloud computing, resource shortage can be minimized. In scheduling process, an over-loaded of host could be occurred. Due to that, the host capacity is less than the application resource requirement, as a result, the application performance can be degraded. To overcome such kind of problem load balancing technique can be one of the solution. In our case, the load balancing for the host is achieved through the virtual machine migration process by taking host utilization by comparing the value set in SLA as percentage value. So, this case can be used to increase the performance of the system. Because the allocation of limited hardware resources can leads to application performance degradation. This problem can be solved by relocating of virtual machines from over-loaded host to under-loaded host with the minimum response time.

User satisfaction level

User satisfaction is how far the services of service provider of cloud can give satisfaction in terms of resources like computation and storage. A clearly and concisely defined SLA increases the customer satisfaction level, as it helps providers to focus on the customer requirements and ensures that the effort is put on the right direction [49].

This can be achieved by SLA based on some QoS parameters. In our case, priority of job by length and utilization of host (for increasing resource usage) are considered. That is high jobs with priority must get first execution and conduct migration when it is necessary based on host usage. Job scheduling is better when it is successful in cloud computing among more

available resources by meeting user satisfaction using QoS taking as SLA parameter. Because, most of the time QoS requirements are commonly formalized in the form of SLAs, which can be determined in terms of the characteristics of the system. As the characteristics of cloud services could be vary for various applications, it is necessary to define the metric for SLA, which can be used to evaluate the SLA delivered by scheduling, which is deployed in cloud service model [50]. In our case, performance degradation due to migrations (PDM and SLA violation Time per Active Host (SLATAH) are metrics of SLA violation.

4.2 System architecture

A system architecture or systems design is the conceptual model that defines the behavior structure, or views of a system. It is a formal description and representation of a system, organized in such a way as to support reasoning about the system's structure and behavior. A system architecture consist of system components and the sub-systems developed, that will work together to implement the overall system. In this Chapter, different parts of components in the architectural design are discussed briefly.

4.2.1 Design components

As it is shown in Figure 4.1, the architecture is composed of six major components namely host monitor, Vm selection, Vm placement, job organizer, SLA module and job scheduler. The demonstration for the aforementioned diagram could be described depending on the architecture. The new model is designed based on the experiment result of those six components. The components have their own specific functions. In the following sections, a detailed description of each component, flow of events and working processes are briefly discussed.

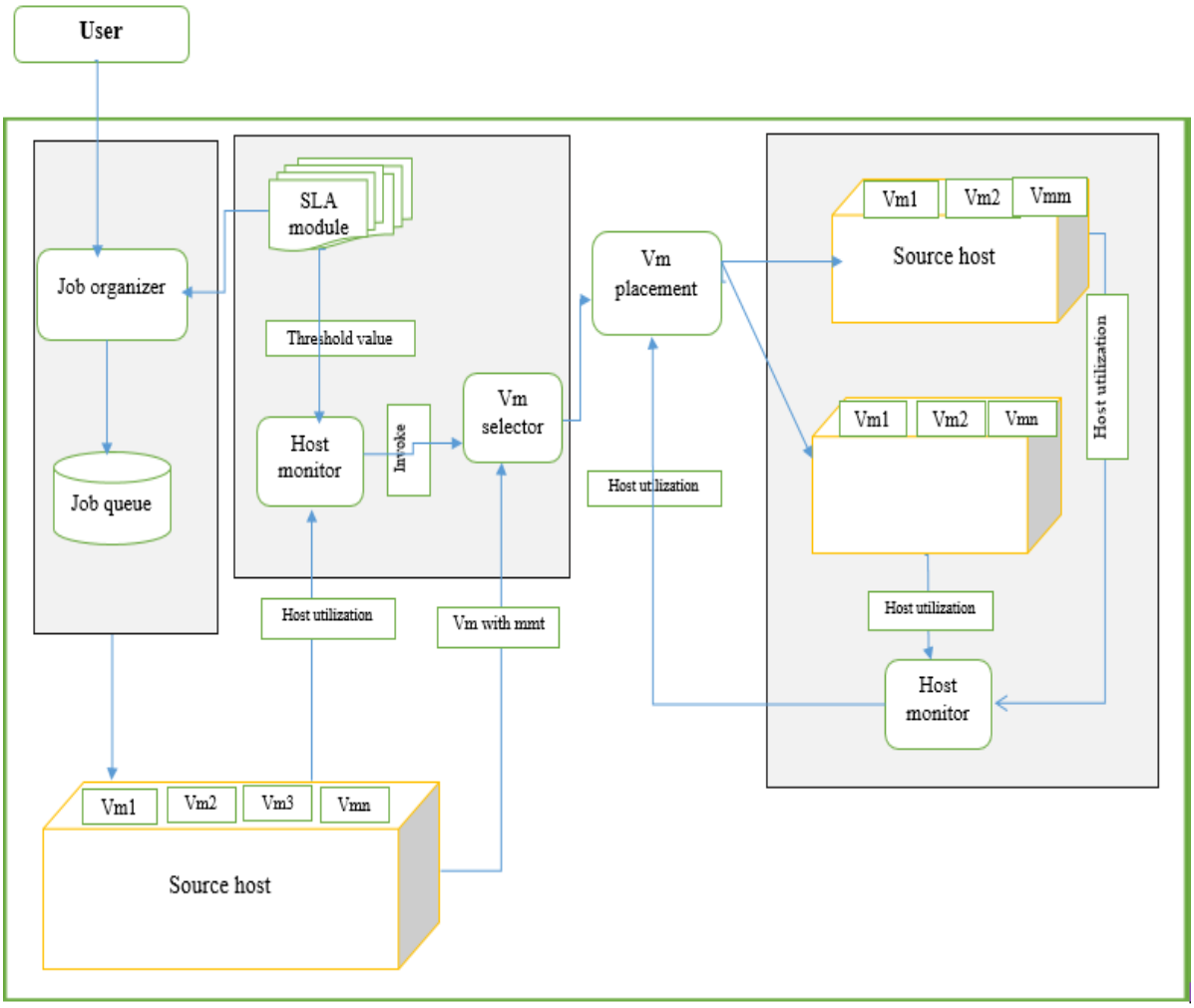

Figure 4.1: Overall system architecture

The working principle of the above architecture is based the agreement which is defined in SLA. The SLA is that the jobs with the shortest length will reside at the first priority and scheduled first and tasks having highest length will have second priority and schedule next by considering utilization of the host. The utilization of host is based on the percentage usage of host and defined in static numeric based threshold value. That is, the incoming jobs are sorted in ascending order, and scheduled to the available resources as first come first serve fashion. During this time, if the usage of source machine is become greater than the value set in threshold value of SLA, the migration process is invoked and Vm placement policy is conducted. The responsibility of each components in the architecture are described below in detail.

i. SLA module

This module contains the agreement between service provider and service consumers. It helps to achieve user satisfaction based on QoS according to the condition of the rules which are set in SLA. In our case, job with small length get first priority to execute and utilization of host based on the threshold value are taken as SLA service parameters.

Job length can express as the mathematical equation shown below [51].

$$TL = Ei * Pc \text{ and } Ei = TL/Pc \quad (1)$$

Where Ei denotes execution time, *TL* represents job length and *Pc* denotes processor capacity. From the above equation, we can understand that job length has a direct influence on execution time. That is if the length of task become greater, the execution time becomes greater. Due to that, by applying the concept of priority of job with small length as a customer need, we sort the job by length in ascending order. If process are executed on the bases of priority, the high priority jobs does not need to wait for long time. Thus, liberate the resources as soon as possible for the other jobs [52].

ii. Job organizer

Users can send various kinds of jobs to the cloud datacenter. Those various jobs may have various lengths. Due to that by taking the job length as a parameter which is defined in SLA, the job organizer can arrange the jobs in ascending order or in descending order. Since, in our case, this module can identify users' job based on job length and arrange those jobs in ascending order according to SLA information, which are set between service provider and service user. After sorting the jobs by their length, the job organizer section sends the jobs to the job queue. The following pseudocode code shows that how jobs are organized based on their length.

```
Input: user Jobs
Output: sorted job based on length in ascending order
     Begin:
          Get jobs with their length
          Compare length of each job
          Set sorted jobs in to queue
```

Algorithm 4. 1: Algorithm to sort jobs

iii. Job Queue

It used to accept jobs as input from job organizer and it stores the arranged jobs before the further scheduling process is conducted.

iv. Host monitor

Host monitor is one of the basic components in the architecture of SLA based job scheduling model for cloud computing environment. This component is responsible for detecting whether a host is over-loaded or under-loaded. This helps to know when the Vms is migrated or not. The method used to know host overloading detection and determine the moment of Vm migration is by setting upper host utilization thresholds value. The threshold value is set in the SLA module of the architecture. If the host utilization exceeds the upper thresholds, it invokes or notify the Vm selection policy. To detect the host is over-loaded or not, it is necessary to know the utilization of CPU, RAM and BW. Therefore, to get the host utilization, taking the product of CPU, RAM and BW from the virtual and physical machines as given by the following equation [53].

$$\boldsymbol{h(u) = \frac{vmj(cpu)usage}{1-pm(cpu)} * \frac{vmj(ram)usage}{1-pm(ram)} * \frac{vmj(bw)usag}{1-pm(bw)}, j = 1, 2, \ldots . m} \quad \textbf{(2)}$$

Where, *h(u)* is utilization of host, *pm(cpu)* physical machine CPU usage, *pm(bw)* is bandwidth usage of physical machine. $vmj(cpu)$usage , $vmj(cpu)$usage is Vm CPU usage and BW usage respectively.

```
1. Input: Utilization of hosts, Threshold value
2. Output: over-loaded host
     Begin:

3.  Get utilization of hosts [53]
4.   Get upper threshold value
5.      If (HU > threshold value)
6. Over-loaded host= host
7. Call Vm selector for selection of vm to migrate
```

Algorithm 4. 2: Algorithm to detect host load

v. Vm selector

This module is responsible for selecting the Vm from the source host to be migrated to the destination host. In a host more than one Vm can be run. During this case, we need to choose migratable Vm when host utilization exceeds the upper threshold value and necessary to set a policy how to select Vm. To do this, minimum migration time (MMT) Vm selection policy is used for the purpose of facilitating response time for users. The MMT policy migrates a Vm

(v) that requires the minimum time to complete a migration relative to the other Vms allocated to the host. The migration time is estimated as the amount of RAM utilized by the Vm divided by the spare network bandwidth available for the host ***j***. Let **Vj** be a set of Vms currently allocated to the host ***j***. After selecting the Vm which has MMT, Vm placement module is called. The MMT policy finds a Vm ***v*** that satisfies conditions of the equation set below [54].

$$\boldsymbol{v} \in \boldsymbol{Vj} | \forall \boldsymbol{a} \in \boldsymbol{Vj}, \frac{\boldsymbol{RAMu(v)}}{\boldsymbol{NETj}} \leq \frac{\boldsymbol{RAMu(a)}}{\boldsymbol{NETj}} \quad (3)$$

```
1. Input: Vms
2. Output: Vm with MMT
   Begin:
3.  Get migratable Vms of hosts
4.  If (RAMu(v)/NETj ≤ RAMu(a)/NETj )     [54]
5.  Get migratable Vm
6.  Send Vm to Vm placement class
```

Algorithm 4. 3: Algorithm to select migratable Vm from source host

vi. Vm placement

After selecting migratable Vm using Vm selection policy of MMT method, it needs the policy to select the host to place the migrated Vm**.** In our case, the Vm placement is based on SLA based approach by taking host utilization. As we discuss in the section of design consideration, the maximum host utilization is 80%. Therefore, the target host can be identified either it is over-loaded or not based on the maximum host utilization value. So the maximum host utilization (HU) value used to determine the capacity of host overloading and help to decide to which host the Vm can be migrated. Therefore, this component can do placement of Vm to the under-loaded host by considering the destination host utilization. It means that, the Vm is migrated to a host with the minimum host usage from the other available target hosts.

```
1. Input:  utilization of host, Vm, threshold value
2. Output: void
3. Begin:
4. get hosts utilization [53]
5. compare HUs with Threshold value
6. get host with minimum utilization
7. Set Vm to target host with minimum utilization
```

Algorithm 4. 4: Algorithm to place Vm target host

vii. Job scheduler

This component is responsible for taking the jobs from the job queue and sends the jobs to the source host. The following pseudocode shows how this module can works.

```
Input:  no of sorted job
Output: void
 Begin:
  Get sorted jobs by length
 While (no of sorted job! =0)
  Send sorted jobs to source hosts
   End while
```

Algorithm 4. 5: Algorithm to schedule the available jobs

4.3 Summary

Job scheduling is the fundamental issue of cloud computing environment and good scheduling must be successful for meeting some QoS parameters. The components such as host monitor, Vm selector, job queue, SLA module and job organizer in the architecture are essential parts for modeling service level agreement based job scheduling for cloud computing environment. Moreover, those components are briefly described. The components in the architecture are aimed to addressing the scheduling performance metric like load balancing and efficient resource utilization.

The new architecture uses host utilization based on threshold value of SLA and length of jobs for processing of scheduling. The jobs are sorted based on their length MIPS for setting priority. After the jobs are organized in ascending order of length, the scheduler sends each job to the source host machine. If the host utilization is over loaded, one of the Vms, which has MMT is migrated to another destination under-loaded host.

Chapter Five: Evaluation and Experimentation Result

This chapter presents the tools and methodology that has been used to implement SLA based job scheduling for cloud computing environment. The prototype is developing based on the architecture represent in Chapter Four. This Chapter also discusses the programming approaches and techniques used to model and manipulate the entities of cloud computing for the proposed method.

5.1 Scope of the prototype

The prototype is designed to meet the basic requirements of the system architecture discussed in Chapter four in Figure 4.1. The components or modules such as Vm selector, Vm placement, host over-load detection and job organizer are implemented. Besides that, the prototype does not visualize components creation and manipulations using user graphical interface. Hence, in the experiment of this simulation, we do not use text-based method.

5.2 Development tools

The tools used for developing prototype for this thesis is described below.

5.2.1 Java Platform, Standard Edition Development Kit (JDK) Version 8

The Java Development Kit (JDK) is a software development environment for building applications and components using Java programming language. The JDK provides tools that are useful to build and test programs written in the Java programming language and running on the Java platform. It includes the Java Runtime Environment (JRE) and development tools, which used to provide an environment to develop user java programs. The JRE provides the minimum requirements for executing a Java application; it consists of the JVM, which enables a computer to run Java programs as well as programs written in other languages that are also compiled to Java byte code, core classes and supporting files. As this research work is Java-based application, the JDK is chosen.

5.2.2 Eclipse IDE

Eclipse is an integrated development environment (IDE) for developing applications using the Java programming language and other programming languages such as C/C++, Python, PERL, etc. The Eclipse platform, which provides the foundation for the Eclipse IDE is composed of plug-ins and is designed to be extensible using additional plug-ins. The Eclipse platform can be used to develop rich client applications, integrated development environments and other tools. Eclipse can be used as an IDE for any programming language for which a

plug-in is available. In this research, we use Eclipse IDE 4.12 release 2019, for writing java programs and integrating the package of cloudsim 3.0.3.

5.2.3 Cloudsim 3.0.3

Cloudsim is an extensible simulation toolkit that allows cloud computing systems and applic ation delivery environments to be modelled and simulated. The cloudsim toolkit supports both systems and behavior for simulation of components of cloud system such as datacenters, Vms and resource provisioning policies. It implements generic application provisioning method for application that can be expanded with ease and minimal effort. At present, it supports modeling and simulation of cloud computing environments consisting of both single and federation of clouds[5]. It also sets out custom interfaces for applying policies and offering Vms allocation techniques under inter-networked cloud computing scenarios [55].

The cloudsim toolkit allows to:

- ✓ Test repeatable and controllable setting for evaluating applications services.
- ✓ Tune the system bottlenecks before deploying applications in an actual cloud.
- ✓ Experiment on virtual infrastructure with various workload mix and resource output scenarios for designing and evaluating adaptive device provisioning techniques.

5.3 Cloudsim Architecture

Cloudsim architecture shows the graphical representation that are concepts of the cloudsim which contains different componenets and their structure.It has core simulation engine, cloudsim layer and user code layer. Figure 5.1 shows the overall cloudSim architecture [55].

[5]Cloud federation is the practice of interconnecting the cloud computing environments of two or more service providers for the purpose of load balancing traffic and accommodating spikes in demand.

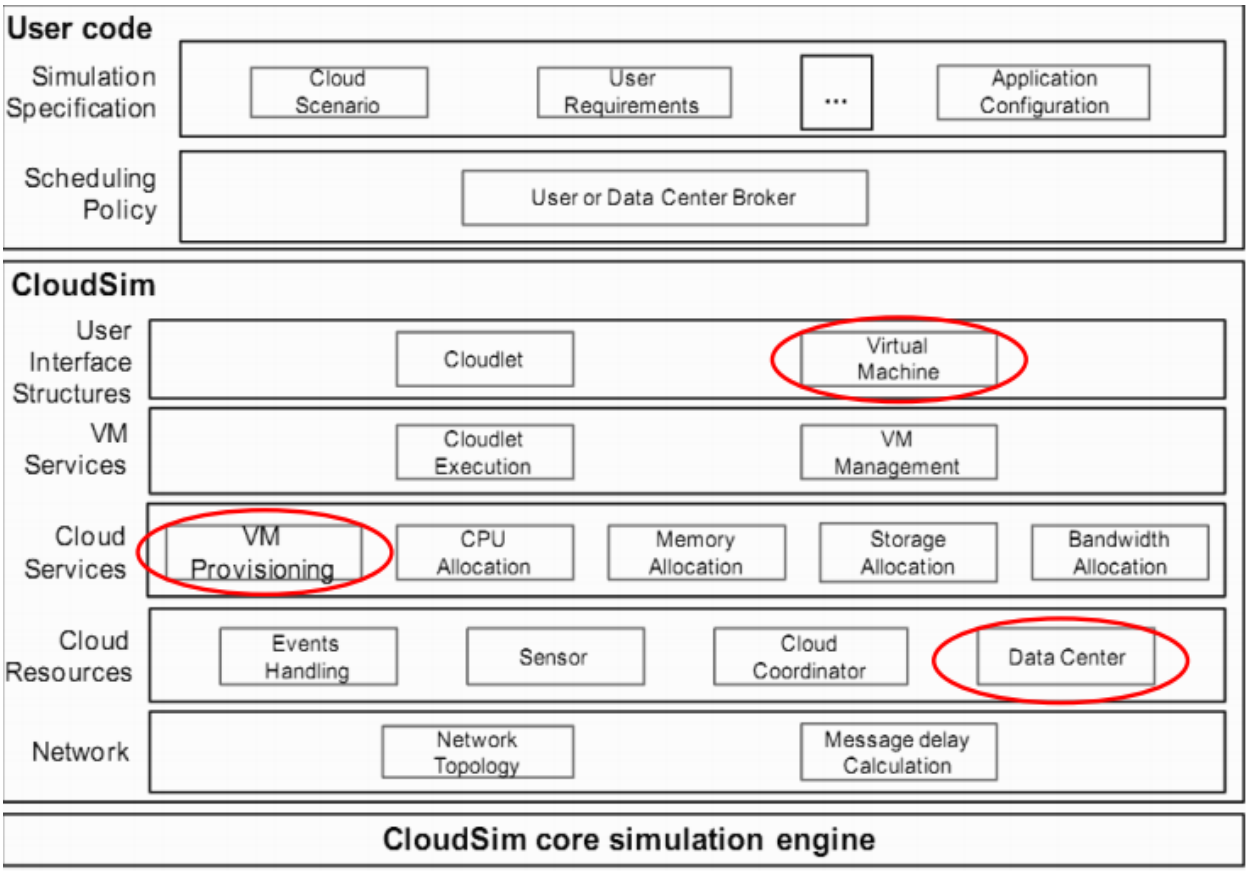

Figure 5.1: layered architecture of cloudSim simulator

As we look in the above Figure, the major cloudsim components are discussed below.

Core simulation engine: The core simulation engine supports the modeling and simulation of virtualized cloud-based datacenter environments, including event queuing and processing the development of cloud system entities such as datacenter, host, Vms, brokers, services, for communication between components and simulation clock management.

CloudSim layer: It provides dedicated management interfaces for Vms, memory, storage and bandwidth. Additionally, it manages the other fundamental issues, such as provisioning of hosts to Vms, managing application execution and monitoring dynamic system state like network topology, sensors, storage characteristics, etc.

User code layer: The user code layer is a custom layer where the user writes their own code to redefine the characteristics of the stimulating environment. Cloudsim consists different kinds of classes.

5.4 Cloudsim Class diagram

Cloudsim has various types of class, which are also the building blocks of the simulator. For the design and implementation of cloud simulations, cloudsim classes have their own special roles [55]. The overall class design diagram for cloudsim [55] is shown in Figure 5.2 below.

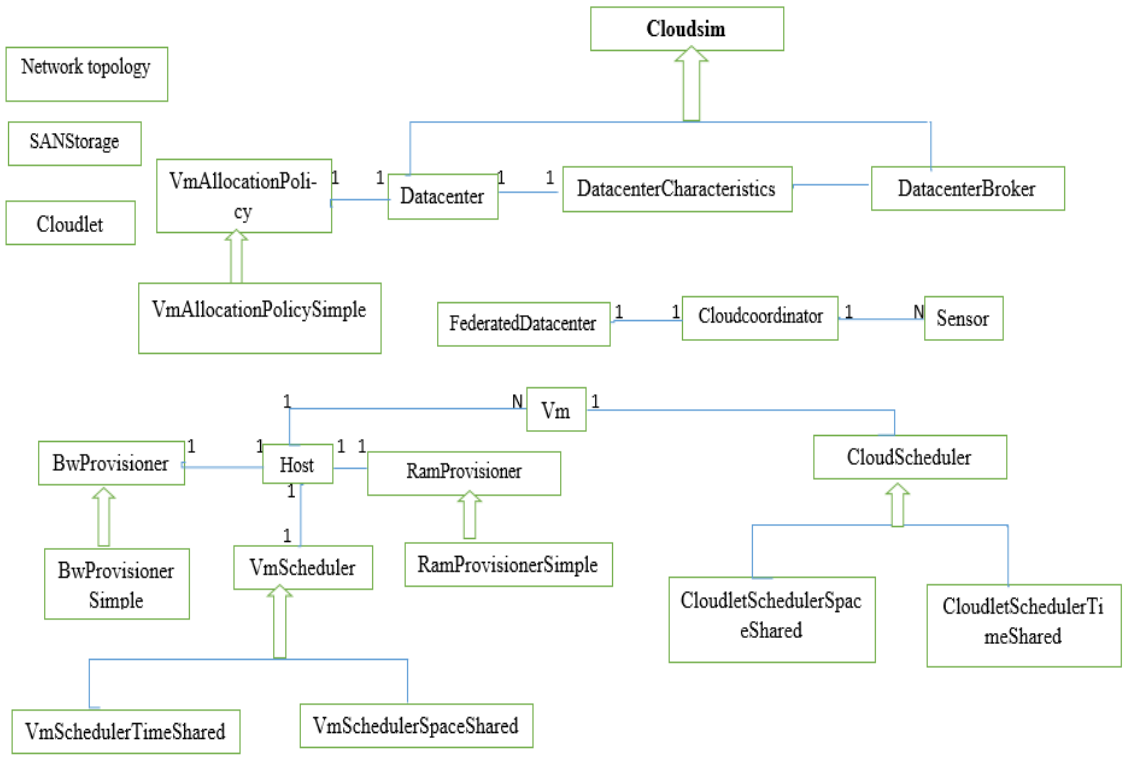

Figure 5. 2: Classes of cloudsim simulator

The role and responsibilities of classes are described below based on [55].

Datacenter:-This class models the core infrastructure level services offered by resource providers in a cloud computing environment. Datacenter class can encapsulates the collection of compute hosts like blade servers that can be either homogeneous or heterogeneous with respect to their resource configurations such as memory, cores and storage.

DatacenterBroker:-This class models a broker, which is responsible for mediating between users and service providers based on the QoS requirements of users and deploys cloud wide service tasks. Via the CIS, the broker working on behalf of users identifies relevant cloud service providers and negotiates with them to assign resources that match users' QoS needs.

DatacenterCharacteristics:-This class contains configuration information of data center resources.

SANStorage:-This class models a network of storage areas typically accessible to cloud-based datacenters to store large chunks of data. SANStorage implements a simple interface, which can be used to simulate the storage and retrieval of any amount of data, subject to network bandwidth that are available at any time.

Host:-This class models a physical resource such as a compute or storage server. It encapsulates important information such as the amount of memory and storage, a list and type of processing cores (to represent a multi-core machine), an allocation of policy for sharing the processing power among Vms, and policies for provisioning memory and bandwidth to the Vms.

VirtualMachine:-This class models an instance of a Vm, whose management is the responsibility of the host component during its life cycle. As mentioned earlier, a host will instantiate multiple Vms simultaneously and assign cores based on predefined processor sharing policies (space-shared, time-shared). Every Vm component has access to a component that stores the characteristics related to a Vm like, accessible memory, processor, storage size, and the Vm's internal provisioning policy that is extended from an abstract component called the CloudletScheduler.

Cloudlet:-It represents the task. This class typically deployed in the datacenters and it is used to models the cloud-based application services (content delivery, social networking and business workflow).

CloudletScheduler:-This abstract class is extended by the implementation of different policies that determine the share of processing power among Cloudlets in a Vm. As described in Figure5.2, there are two types of provisioning policies called space-shared (CloudletSchedulerSpaceShared) and time-shared (CloudletSchedulerTimeShared).

BwProvisioner:-This is an abstract class modeling the bandwidth provisioning policy for Vms deployed on the host component. This component has the purpose of allocating network bandwidths to the collection of competing Vms deployed around the datacenter. Cloud system

developers and researchers can expand this class to represent the needs of their applications using their own policies like that of priority and QoS.

VmProvisioner:-This is an abstract class that can demonstrates the provisioning strategy for Vms memory allocation. This component models policies for the allocation to the competing Vms of physical memory spaces. The VmProvisioner's main functionality is to pick available host in a datacenter that meets the requirement of memory, storage and availability for a Vm deployment.

VmAllocationPolicy:-This is an abstract class implemented by a host component that can capable of modeling the policies (space-shared, time-shared) needed for Vms processing power allocations. The chief functionality of the VmAllocationPolicy is to select the available host in a data center that meets the memory, storage, and availability requirement for a Vm deployment.

CloudCoordinator:-This abstract class extends a Cloud-based data center to the federation. It is responsible for periodically monitoring the internal state of data center resources and based on that it undertakes dynamic load-shredding decisions.

NetworkTopology:-This class contains the information for inducing network behavior (latencies) in the simulation. It stores the topology information, which is generated using the BRITE topology generator.

Sensor:-This interface must be implemented to instantiate a sensor component that can be used by a CloudCoordinator for monitoring specific performance parameters (energy-consumption, resource utilization).

VmScheduler:-This is an abstract class implemented by a Host component that models the policies (space-shared, time-shared) required for allocating processor cores to VMs. The functionalities of this class can easily be overridden to accommodate application specific processor sharing policies.

5.5 Prototype development

Prototype is developed to implement the proposed scheduling algorithm. This prototype implements only the job scheduler and Vm migration.

Datacenter

In real world, datacenter is a facility used to as house of computer systems or servers and associated components such as network communications and storage systems. This includes backup or backup power supplies, backup connections to data transmission, environmental controls such as fire suppression air conditioning and numerous monitoring devices. Datacenters have evolved significantly, improving efficiency and increasing scale to adopt techniques such as virtualization, cloud computing, mobile and internet of things application.

Datacenter is used to model the core services at the system level of a cloud infrastructure. It consists of a set of hosts, which manage a set of Vms whose tasks are to handle "low level" processing and at least one datacenter must be created to start the simulation. To create datacenter, *org.cloudbus.cloudsim.datacenters* package is responsible in cloudsim framework. For modeling datacenter, hosts, datacenter brokers, Vms are the necessary entities. Table 5.1 shows the overall datacenter design for designing datacenter for the proposed model.

Table 5.1: Sample of datacenter design

Host	RAM(MB)	MIPS	BW	Secondary memory (MB)	Cores
Host 1	40000	1000	16000	1000000	1
Host 2	30000	2000	1600	1000000	1
Host 3	20000	1000	1500	1000000	1
Host 4	50000	3000	16000	1000000	2
Host 5	10000	4000	1000	1000000	1
Host 6	10000	4000	1000	1000000	1
Host 7	30000	2000	2000	1000000	1
Host 8	20000	1000	1000	1000000	1
Host 9	10000	3000	2000	1000000	1
Host 10	30000	10000	25000	1000000	1

As it is shown in Table 5.1, the resources in this scheduling model are located in such way. Hence, this model is used to show the resources structure, which are given in datacenter.

Annex A contains line of code which used to create a host lists in the datacenter in cloudsim simulation for the proposed model.

SLA module

This component contains the agreement information between the provider and the service user. As we describe in Chapter four, it contains the percentage threshold value for detecting host utilization and notifies the job are executed in ascending order based on their length. Due to this, the user could be satisfied based on their agreement and if it is not done well the SLA violation is also calculated for the purpose of penalty.

During the experiment, we define that the SLAs are delivered when utilization of hosts are not greater than 80%. In addition, performance degradation due to migration is measured. Because, migration of virtual machine is said to be good, it is not only trying to move a virtual machine from one host to another as fast as possible, but also needs to minimize its side effects. In order to measure such types of strategy in SLA, service degradation is the one metric. Service degradation indicates how the service running in the migrated virtual machine is affected by the migration. It is called performance degradation due to migrations (PDM). It can be measured by the changes of throughput and response time [56].

Depending on this, we have taken PDM as a metrics for measuring the level of SLA violations. The other one is SLA violation Time per Active Host (SLATAH). This is the percentage of time during the active hosts have experienced the host utilization of 80%. The notion behind the SLATAH is the observation that if a host serving applications is experiencing 80% utilization, the performance of the applications is limited by the host capacity helps to know SLA violation. Therefore, Vms are not being provided with the required performance level. This can be calculated based on the following equation [50].

$$SLATH = \frac{1}{N}\sum_{I=1}^{N} Tsi/Tai \qquad (4)$$

$$PDM = \frac{1}{M}\sum_{j=1}^{M} \frac{Cdj}{Crj} \qquad (5)$$

Where *N* is the number of hosts, *Tsi* is the total time during which the host *i* has the utilization is above 80% leading to an SLA violation. *Tai* is the total of the host i being in the active state (serving Vms). *M* is the number of Vms, *Cdj* is the estimate of the performance degradation of the *Vmj* caused by migrations and Crj is the total CPU capacity requested by the *Vmj* during

its lifetime. In our experiments, we set *Cdj* as 80% of the host utilization in in million instruction per second (MIPS) during all migrations of the *Vmj*. Both the metrics of SLATAH and PDM could have the same significance which characterized the level of SLA violations by the infrastructure of the service. Therefore, we have taken both metrics that encompass both performance degradation due to host overloading and Vm migrations and the metric of SLA Violation (SLAV) can be calculated as shown in the following equation in [50].

$$SLAV = SLATAH * PDM \quad (6)$$

Where, SLAV is Service Level Agreement violation, SLATAH is SLA violation Time per Active Host and PDM is performance degradation due to migration. Annex B shows prototype code line of this module.

Job organizer

The job organizer component accept jobs with different length from user and it arranges the jobs in ascending order as we discussed in Chapter four. Therefore, in this prototype, it contains the lines of programs that can arrange jobs based on length as shown in Annex C.

Vm selector

In this phase, the component consists line of instruction for selecting Vm, which has minimum migration time as mentioned in Chapter four in equation (3). The minimum migration time policy is selected for facilitating response time for users. For this, Annex D shows the implementation prototype sample of Vm selection for enabling migration process.

Vm placement

The phase of this component implementation focuses on placing Vm in the target host, which is migrated from source host to targeted hosts. As discussed in the previous chapter (Chapter 4) the placement is decided based on the utilization of host comparing with threshold value. Annex E notifies the implementation sample of Vm placement.

Host overload detection

In this case, the load of hosts must be identified. To do that, host overload detection prototype is implemented as shown in Annex F. Host overload detection is used to detect weather the host is under-loaded or over-loaded based on the threshold value. The host utilization is calculated by mathematical formula in equation (2) Chapter four.

Test result

The prototype of the proposed model is developed on a machine which has Intel(R) core(TM) i5 Processor, 2.4 GHz of speed with memory 8 GB RAM, windows 10 platform and using cloudsim 3.0.3 simulator. The cloudsim toolkit supports modeling of cloud system components such as data centers, host, virtual machines, resource provisioning policies, etc. The experiment is held on the cloud models shown in Tables 5.2 and Table 5.3. Since, Table 5.2 and Table 5.3 shows test result that we have get the output of the prototype simulation in the experiment.

Table 5. 2: Table that show simulation result of sorted job

Cloudlet ID	Status	DC ID	Host ID	Host PES	VM ID	Cloudlet Length	Execution time in seconds
0	Success	1	0	4	0	2000	2
8	Success	1	4	8	8	2000	2
7	Success	1	3	7	7	4000	4
2	Success	1	1	5	2	9000	10
5	Success	1	3	7	5	9000	10
3	Success	1	2	6	3	10000	11
4	Success	1	2	6	4	10000	11
1	Success	1	1	5	1	16000	17
9	Success	1	4	8	9	20000	28
6	Success	1	3	7	6	30000	30

During the simulation process we have get the result of SLAV, SLAPDM and SLATAH. Table 5.3 shows that the simulation result mainly related to SLA metrics under the given number of hosts and Vms.

Table 5. 3: Table that show the metrics value on used number of hosts and Vms

No of hosts	10
No of vms	10
SLAV	2.502%
PDM	0.025%
SLATAH	10.01%

During the running of cloudlets or jobs on the simulation we get two scenarios. Those scenarios are defined in graph form as shown below. Figure5.8 shows that running of jobs without sorting by taking jobs with small length has first priority. In this case jobs which have small length do not have much more waiting time to execute. The following graph shows the result of the proposed model.

Figure 5. 3: Result of SLA based job scheduling with sorting by job length

As we look in Table .4 and Figure 5.9, jobs are executed without sorting. From this, we understand that jobs with small length have chance for getting delay because of waiting jobs which have greater length.

Table 5. 4: Table that shows execution result of jobs without sorting

Cloudlet ID	Status	DC ID	Host ID	Host PES	VM ID	job length	execution time
0	Success	1	0	1	0	2000	2
1	Success	1	4	1	8	16000	17
2	Success	1	3	1	7	9000	10
3	Success	1	1	1	2	10000	11
4	Success	1	3	1	5	10000	11
5	Success	1	2	1	3	9000	10
6	Success	1	2	1	4	30000	31
7	Success	1	1	1	1	4000	4
8	Success	1	4	1	9	2000	2
9	Success	1	3	1	6	20000	28

Figure 5. 4: Result of SLA based scheduling without sorting of jobs

Chapter Six: Conclusion and Future work

This chapter recaps the major findings in this research work .It also summarized the key concepts of cloud computing, SLA and scheduling in this research work. Additionally, the main contributions of the proposed job scheduling algorithm and future works are pointed.

6.1 Conclusion

Cloud computing can assign remote services with a user's data, computation and software. It is a model, which provides on-demand access to shared pool computing resources like servers, storage, networks, applications and services. It uses network of distant servers hosted on the internet to store, manage and process data, rather than a local server. Virtualization is the strategic aspect for cloud computing that enables multiple customers to use the physical resources.

Cloud computing model can be seen from two sides. The deployment model, which refers to where the cloud is located and for what purpose is applied and service model, which refers what type of services are provided to users. The deployment model categorized in to four parts namely public cloud, private cloud, hybrid cloud and community cloud. The most known service model of cloud computing is SaaS, PaaS and IaaS.

Virtualization makes handling the resources simple for the cloud computing world. One of the most significant aspects of Vm technology is migration of the Vm. Vm migration helps to balance the load, resource management and reducing energy usage in scheduling of cloud resources. Scheduling is the basic technology for cloud computing for effective management and access of cloud resource. Scheduling is a method of transferring sets of jobs to a set of Vms or allocating Vms to operate on the resources available to satisfy the needs of users.

This study has shown that the general concept of cloud computing and scheduling conducted by different researchers. It identified job scheduling by taking major contributing factors for the SLA of the cloud party. As cloud computing technology, provide services to customers and businesses, allowing organizations to become more flexible. As well as there are different types of resources and various cloud service customer needs and SLA between cloud parties in cloud environment. Due to that, there is not a uniform standard for job scheduling and it is necessary to conduct job scheduling based on the agreements of users and providers. The proposed study contains Vm selector, Vm placement, host monitor and job organizer as basic

component of scheduling. The result of this study is examined by cloud simulation framework called cloudsim.

6.2 Contributions of the Research

In this research work, the main involvement is increase the resource utilization for user and achieving user satisfaction level. Accordingly, we can determine the host load for better resource utilization based on the percentage usage of host and invoke Vm migration when it is necessary. Hence, this study considered and implemented various components to achieve this goal.

6.3 Future work

The SLA based scheduling in cloud computing environment proposed in this research could be used in various cloud vendor organizations. Moreover, various research works will use the methodology and findings of this research work by taking different QoS parameters like deadline and security as an agreement of cloud parties.

References

[1] S. Naidila and K. Dilip, "Cluster Grid and Cloud Computing: A Detailed Comparison," in *The 6th International Conference on Computer Science & Education (ICCSE)*, Singapore, 2011.

[2] C. S. Yeo, B. Rajkumar, P. Hossein, E. Rasit, G. Peter and S. Frank, "Cluster computing: High-performance, high-availability, and high-throughput processing on a network of computers," in *In Handbook of nature-inspired and innovative computing*, Boston, Springer, 2006, pp. 521-551.

[3] B. Rebecca M. and S. Acting, National Institute of Standards and Technology, USA: U.S. Department of Commerce, 2011.

[4] Sonia, "Task scheduling in cloud computing," *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, vol. 4, no. 6, 2015.

[5] M. T. Nidal, A. T. Anas and R. M. Shadi, "Cloud Computing Challenges and solutions," *International Journal of Computer Networks and Communications*, vol. 5, 2013.

[6] F. Mohammadi, S. Jamali and M. Bekravi, "Survey on Job Scheduling algorithms in Cloud computing," *International Journal of Emerging Trends & Technology in Computer Science(IJETTCS)*, vol. 3, no. 2, pp. 151-154, 2014.

[7] C. Seema and S. Shirwaikar, "Predictive Modeling of Service Level Agreement Parameters for Cloud Services," *International Journal of Next-Generation Computing*, 2016.

[8] W. Usman, F. G. Khan and S. Sajid, "Service Level Agreement in Cloud Computing: A survey and SLA violation," *International Journal of Computer Science and Information Security* , vol. 14, no. 6, June 2016.

[9] F. Azimzadeh and F. Biaban, "Multi-Objective Job Scheduling Algorithm in Cloud Computing Based on Reliability and Time," in *International Conference on Web Research*, 2017.

[10] P. Pankesh, R. Ajith and S. Amit, "Service Level Agreement in Cloud Computing," in *International Conference on Web Research*, 2014.

[11] Z. Yao, I. Papapanagiotou and D. C. Robert, "SLA-aware Resource Scheduling for Cloud Storage," in *IEEE 3rd international conference cloud networking*, 2014.

[12] M. I. Biswas, P. Gerard, M. Sally, M. Philip and S. Bryan, "SLA-Based Scheduling of Applications for Geographically Secluded Clouds," in *International Conference and Workshop on the Network of the Future (NOF) IEEE*, 2014.

[13] S. K. Panda and K. J. Prasanta, "SLA-based task scheduling algorithms for heterogeneous multi-cloud environment," *The Journal of Supercomputing,* vol. 73, no. 6, pp. 2730-2762, 2017.

[14] M. S. Gill and R. K. Bawa, "Service level agreement based fault tolerant workload scheduling in cloud computing environment," *International Journal of Grid Computing and Applications*, vol. 7, 2016.

[15] D. M.Deebiga, P.Gowthaman and B.Krishnakumar, "SLA-based Virtual Machine Scheduler In Cloud Environment,," *International Journal of Advanced Research Trends in Engineering and Technology (IJARTET),* vol. 4, no. 14, 2017.

[16] R. B. Kaippilly and S. Philip, "Service-level agreement–aware scheduling and load balancing of tasks in cloud," 2018.

[17] S. N. Samreen, N. K. Valmik, S. Salve and P. N. Khan, "Introduction to Cloud Computing," *International Research Journal of Engineering and Technology,* vol. 5, no. 2, pp. 785-788, 2018.

[18] K. S. Krutika, V. U. Vadiya and R. Jhaveri, "Survey Paper on Security in Cloud Computing: Bibliographic Analysis," *Circulation in Computer Science,* vol. 1, no. 2, pp. 19- 23, 2016.

[19] P. Anupama, "Cloud Computing Services," *International Journal of Computer Applications,* vol. 46, no. 3, 2012.

[20] T. Swathi, K.Srikanth and R. S.Raghunath, "Virtualization in cloud computing," *International Journal of Computer Science and Mobile Computing*, vol. 3, no. 5, p. 540 – 546, 2014.

[21] G. J. Mirobi and L. Arockiam, "Service Level Agreement In Cloud Computing Overview," in *International Conference on Control,Instrumentation, Communication and Computational Technologies (lCCICCT)*, 2015.

[22] A. Rashid and A. Chaturvedi, "Cloud computing characteristics and services: a brief review," *International Journal of Computer Sciences and Engineering,* vol. 7, no. 2, pp. 421-426, 2019.

[23] T. Huaglory, "Cloud Computing Architectures," *School of Engineering and Built Environment Glasgow Caledonian University* , 2011.

[24] A. M. Chandrashekhar and Shashikumar, "Cloud Computing Service and Deployment Models," *International Journal for Research in Applied Science & Engineering Technology (IJRASET),* vol. 5, no. 6, pp. 1083-1088, 2017.

[25] M. Alam and B. Sakib, "Cloud Computing Architecture Platform and Security Issues: A Survey," *World Scientific News 86,* vol. 5, no. 6, pp. 253-264, 2017.

[26] T. Diaby and B. B. Rad, "Cloud Computing A review of the Concepts and Deployment Models," *International Journal of Information Technology and Computer Science,* vol. 6, no. 9, pp. 50-58, 2017.

[27] S. Goyal, "Public vs Private vs Hybrid vs Community - Cloud Computing: A Critical Review," *International Journal of Computer Network and Information Security,* vol. 6, no. 3, 2014.

[28] Geeta and S. Prakash, "Role of Virtualization Techniques in cloud computing environment," in *Advances in Computer Communication and Computational Sciences,* Springer, Singapore, 2019.

[29] A. S. Pal and K. P. B.Prasant, "Classification of Virtualization Environment for Cloud Computing," *Indian Journal of Science and Technology,* vol. 6, no. 1, pp. 3965-3971, 2013.

[30] P. Kaur and A. Rani, "Virtual Machine Migration in Cloud Computing," *International Journal of Grid Distribution Computing,* vol. 8, pp. 337-342, 2015.

[31] M. H. hirvani, A. M. Rahmani and S. Amir, "A survey study on virtual machine migration and server consolidation techniques in DVFS-enabled cloud datacenter: Taxonomy and challenges," *Journal of King Saud University Computer and Information Sciences,* vol. 32, no. 3, pp. 267-286, 2018.

[32] K.Padmaja, R. Seshadri and P. Anusha, "Different Scheduling Algorithms in Types of Clouds," *International Journal of Computer Science Trends and Technology (IJCST),* vol. 4, no. 5, 2016.

[33] S. P. M. Padmavathi, S. M. Basha and P. Srinivas, "A Survey on Scheduling Algorithms in Cloud Computing," *Journal of Computer Engineering (IOSR-JCE),* vol. 16, no. 4, pp. Padmavathi, Srinivas Pothapragada M., Shaik Mahabbob Basha, and Srinivas Pothapragada, Aug. 2014.

[34] T. Ma, YaChu, Z. Licheng and A. Otgonbayar, "Resource Allocation and Scheduling in Cloud Computing policy and algorithm," *IETE technical review,* vol. 31, no. 1, pp. 4-16, 2014.

[35] A. Wadhonkar and D. Theng, "A Task Scheduling Algorithm Based on Task Length and Deadline in Cloud Computing," *International Journal of Scientific & Engineering Research,* vol. 7, no. 4, pp. 1905-1909, 2016.

[36] M. Choudhary and K. P. Sateesh, "A dynamic optimization algorithm for task scheduling in cloud environment," *International Journal of Engineering Research and Applications (IJERA)* , vol. 2, no. 3, pp. 2564-2568, 2012.

[37] N. Almezeini and A. Hafez, "Review on Scheduling in Cloud Computing," *International Journal of Computer Science and Network Security,* vol. 18, 2018.

[38] Y. Chawla and M. Bhonsle, "Study on Scheduling Methods in Cloud Computing," *International Journal of Emerging Trends and Technology in Computer Science (IJETTCS),* vol. 1, no. 3, pp. 12-17, 2012.

[39] D. Kaur and T. Sharma, "Scheduling Algorithms in Cloud Computing," *International Journal of Computer Application,* 2019.

[40] S. K. Panda, A. Pratik and P. M. Durga, "Intermediate mode scheduling in computational grid," *nternational Conference on Green Computing Communication and Electrical Engineering (ICGCCEE),* pp. 1-6, 2014.

[41] N. Soltani, S. Behzad and B. Behrang, "Heuristic Algorithms for Task Scheduling in Cloud Computing: A Survey," *International Journal of Computer Network and Information Security* , vol. 11, no. 8, 2017.

[42] F. Teng, "Ressource allocation and schelduling models for cloud computing," in *PhD dissertation,* Paris, 2011.

[43] S. Pandey, A. Kumar and C. Jha, "Need of SLA Parameters in Cloud Environment Evaluation," *International Journal of Computer Science & Engineering Technology (IJCSET),* vol. 7, no. 12, 2016.

[44] C. Yun, J. M. Hyun, H. Hakan and T. Junichi, "SLA-tree: a framework for efficiently supporting SLA-based decisions in cloud computing," in *In Proceedings of the 14th International Conference on Extending Database Technology*, 2011.

[45] B. S. Rajeshwari, M. Dakshayini and H. S. Guruprasad, "Service Level Agreement based Scheduling Techniques in Cloud: A Survey," *International Journal of Computer Applications,* vol. 132, 2015.

[46] S. B. Dash, H. Saini, T. C. Panda and A. Mishra, "Service Level Agreement Assurance in Cloud Computing: A Trust Issue," *International Journal of Computer Science and Information Technologies,* vol. 5, 2014.

[47] S. Ferretti, G. Vittorio, P. Fabio, P. Michele and T. Elisa, "QoS - Aware Clouds," in *IEEE 3rd International Conference on Cloud Computing*, 2010.

[48] S. Mubeen, A. Sara Abbaspour, V. Alessandro, A. Mohammad, H. Pei-Breivold and B. Moris, "Management of Service Level Agreements for Cloud Services in IoT: A Systematic Mapping Study," IEEE Access 6, 2017, pp. 30184-30207..

[49] L. Wu and B. Rajkumar, "Service level agreement (SLA) in utility computing systems," *In Performance and dependability in service computing: Concepts, techniques and research directions*, pp. 1-25, 2012.

[50] A. Beloglazov and B. Rajkumar, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers," *Cloud Computing and Distributed Systems Laboratory,* vol. 24, no. 13, pp. 1397-1420, 2012.

[51] P. Krishnadoss and J. Prem, "Task Scheduling Algorithm in Cloud Computing Environment," *International Journal of Intelligent Engineering and Systems,* vol. 11, 2018.

[52] N. Erraji and F. Benabbou, "Priority Task Scheduling Strategy for Heterogeneous Multi-Datacenters in Cloud Computing," *International Journal of Advanced Computer Science and Applications (IJACSA),* vol. 8, no. 2, 2017.

[53] A. V. Sajitha and A. C. Subhajini., "Dynamic VM Consolidation Enhancement for Designing and Evaluation of Energy Efficiency in Green Data Centers Using Regression Analysis," *International Journal of Engineering & Technology,* vol. 7, no. 3.6, pp. 179-186, 2018.

[54] S. Namdev, S. Neelam and A. K. Rai, "Improved Minimum Migration Time VM Selection Policy for Cloud Data Center," *International Journal of Application or Innovation in Engineering & Management (IJAIEM),* vol. 4, no. 4 , April 2015.

[55] R. N. Calheiros, R. Ranjan, B. Anton, C. A. Rose and B. Rajkumar, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and experience,* vol. 41, no. 1, pp. 23-50, 2010.

[56] F. Zhang, L. Guangming, X. Fu and Y. Ramin, "Survey on Virtual Machine Migration:Challenges, Techniques and Open Issues," *IEEE Communications Surveys & Tutorials,* vol. 20, no. 2 , pp. 1206-1243, 2017.

Annex A: Sample code to show creating hosts in datacenter

```
public static List<PowerHost> createHostList(int hostsNumber) {
    List<PowerHost> hostList = new ArrayList<PowerHost>();
    for (int i = 0; i < hostsNumber; i++) {
        int hostType = i % Constants.HOST_TYPES;

        List<Pe> peList = new ArrayList<Pe>();
        for (int j = 0; j < Constants.HOST_PES[hostType]; j++) {
            peList.add(new Pe(j, new PeProvisionerSimple(Constants.HOST_MIPS[hostType])));
        }

        hostList.add(new PowerHostUtilizationHistory(
                i,
                new RamProvisionerSimple(Constants.HOST_RAM[hostType]),
                new BwProvisionerSimple(Constants.HOST_BW),
                Constants.HOST_STORAGE,
                peList,
                new VmSchedulerTimeSharedOverSubscription(peList),
                Constants.HOST_POWER[hostType]));
    }
    return hostList;
```

Annex B: Sample code to show SLA violation

```
protected static double getSlaTimePerActiveHost(List<Host> hosts)
{
    double slaViolationTimePerHost = 0;
    double totalTime = 0;

    for (Host _host : hosts) {
        HostDynamicWorkload host = (HostDynamicWorkload) _host;
        double previousTime = -1;
        double previousAllocated = 0;
        double previousRequested = 0;
        boolean previousIsActive = true;

        for (HostStateHistoryEntry entry : host.getStateHistory()) {
            if (previousTime != -1 && previousIsActive) {
                double timeDiff = entry.getTime() - previousTime;
                totalTime += timeDiff;
                if (previousAllocated < previousRequested) {
                    slaViolationTimePerHost += timeDiff;
                }
            }

            previousAllocated = entry.getAllocatedMips();
            previousRequested = entry.getRequestedMips();
            previousTime = entry.getTime();
            previousIsActive = entry.isActive();
        }
    }

    return slaViolationTimePerHost / totalTime;}
```

Annex C: Sample code to show for sorting of jobs based on length

```
public static List<Cloudlet> createCloudletList(int brokerId, int cloudletsNumber) {
    List<Cloudlet> list = new ArrayList<Cloudlet>();

    final double SIMULATION_LIMIT = 24 * 60;

    long fileSize = 300;
    long outputSize = 300;
    long seed = RandomConstants.CLOUDLET_UTILIZATION_SEED;
    UtilizationModel utilizationModelNull = new UtilizationModelNull();
      int[] CLOUDLET_LENGTHS = {2000, 4000, 10000, 16000, 2000, 30000, 20000,6000,7000,9000};

       int temp;

       for(int i=0; i<30;i++) {


           for(int j=i+1;j<30;j++) {


               if(CLOUDLET_LENGTHS[i]>CLOUDLET_LENGTHS[j]) {
                   temp=CLOUDLET_LENGTHS[i];
                   CLOUDLET_LENGTHS[i]=CLOUDLET_LENGTHS[j];
                   CLOUDLET_LENGTHS[j]=temp;

                   CLOUDLET_LENGTHS[j]= 250*(int) (CLOUDLET_LENGTHS[j]*SIMULATION_LIMIT);

               }
           }

       }
```

Annex D: Sample code of vm selector for migration

```
public class PowerVmSelectionPolicyMinimumMigrationTime extends PowerVmSelectionPolicy {

    /* (non-Javadoc)
    @Override
    public Vm getVmToMigrate(PowerHost host) {
        List<PowerVm> migratableVms = getMigratableVms(host);
        if (migratableVms.isEmpty()) {
            return null;
        }
        Vm vmToMigrate = null;
        double minMetric = Double.MAX_VALUE;
        for (Vm vm : migratableVms) {
            if (vm.isInMigration()) {
                continue;
            }
            double metric = vm.getRam();
            if (metric < minMetric) {
                minMetric = metric;
                vmToMigrate = vm;
            }
        }
        return vmToMigrate;
    }
```

Annex E: Sample code for vm placement

```
public class PowerVmAllocationPolicyMigrationStaticThreshold extends
PowerVmAllocationPolicyMigrationAbstract {
  private double utilizationThreshold = 0.8;
     public PowerVmAllocationPolicyMigrationStaticThreshold(
      List<? extends Host> hostList,
      PowerVmSelectionPolicy vmSelectionPolicy,
      double utilizationThreshold) {
  super(hostList, vmSelectionPolicy);
  setUtilizationThreshold(utilizationThreshold);
 }
  @Override
   protected boolean isHostOverUtilized(PowerHost host) {
  addHistoryEntry(host, getUtilizationThreshold());
  double totalRequestedMips = 0;
  for (Vm vm : host.getVmList()) {
      totalRequestedMips += vm.getCurrentRequestedTotalMips();
  }
  double utilization = totalRequestedMips / host.getTotalMips();
  return utilization > getUtilizationThreshold();
}

  protected void setUtilizationThreshold(double utilizationThreshold) {
  this.utilizationThreshold = utilizationThreshold;
 }
```

Annex F: Sample code for host overload detection

```
private UtilizationModelDynamic createHostUtilizationModel(double
initialHostUsagePercent) {
        return createHostUtilizationModel(initialHostUsagePercent,
initialHostUsagePercent);
    }

    private UtilizationModelDynamic createHostUtilizationModel(double
initialHostUsagePercent, double maxHostUsagePercentage) {
        if(maxHostUsagePercentage < initialHostUsagePercent){
            throw new IllegalArgumentException("Max Host usage must be equal or
greater than the initial Host usage.");
        }

        initialHostUsagePercent = Math.min(initialHostUsagePercent, 1);
        maxHostUsagePercentage = Math.min(maxHostUsagePercentage, 1);
        UtilizationModelDynamic um;
        if (initialHostUsagePercent < maxHostUsagePercentage) {
            um = new UtilizationModelDynamic(initialCpuUsagePercent)
                        .setUtilizationUpdateFunction(this::getHostUsageIncrement);
        } else {
            um = new UtilizationModelDynamic(initialHostUsagePercent);
        }

        um.setMaxResourceUtilization(maxHostUsagePercentage);
        return um;
    }
```

Declaration

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

Declared by:

Name: Tilksew Shiferaw

Signature: _____

Date: _____

Confirmed by advisor:

Name: _____

Signature: _____

Date: _____

Place and date submission: Addis Ababa University, 7 October 2020