

Efficient Cloud Workload Management Framework

*Thesis submitted in partial fulfillment of the requirements
for the award of degree of*

Master of Engineering
in
Software Engineering

Submitted By
Sukhpal Singh
(Roll No. - 801131024)

Under the supervision of:
Dr. Inderveer Chana
Associate Professor



COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

July 2013

Certificate

I hereby certify that the work which is being presented in the thesis entitled, "*Efficient Cloud Workload Management Framework*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in Software Engineering submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Dr. Inderveer Chana* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.



(Sukhpal Singh)


This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.


(Dr. Inderveer Chana) 12/07/13

Associate Professor
Computer Science and Engineering Department
Thapar University, Patiala

Countersigned by:


(Dr. Maninder Singh)
Head
Computer Science and Engineering Department
Thapar University
Patiala


(Dr. S. K. Mohapatra)
Dean (Academic Affairs)
Thapar University
Patiala

Acknowledgement

First of all, I am thankful to God for his blessings and showing me the right direction.

With His mercy, it has been made possible for me to reach so far.

At the outset, I would like to express my appreciation to Dr. Inderveer Chana for her advice during my master's research endeavor for the past one year. As my supervisor, she has constantly forced me to remain focused on achieving my goal. Her observations and comments helped me to establish the overall direction of the research and to move forward with investigation in depth. I thank her for providing me with the opportunity to work with a talented team of researchers. Her constant pursuit for perfection, guidance and valuable suggestions have guided me at every step of my career. Particularly, I value her respect towards professionalism and the desire to excel higher and higher. Apart from research expertise, she has made me competent in organizing and managing research activities such as international conferences, workshops and software demonstrations.

I am also thankful to Dr. Maninder Singh, Head, Computer Science and Engineering Department for his motivation, kind help and cooperation. I would like to thank administrative and technical staff members of the Computer Science and Engineering Department who have been kind enough to advise and help in their respective roles.

I would like to express my sincere thanks to Dr. RajKumar Buyya (University of Melbourne, Australia) for generously sharing his time and knowledge in our cooperative work on resource scheduling in Cloud Computing.

I am grateful to numerous local and global "peers" who have contributed towards shaping this thesis.

Last but not the least I am highly grateful to all my family members for their inspiration and ever encouraging moral support, which enables me to pursue my studies. All my friends and family have been a source of inspiration in my life.

Sukhpal Singh

Cloud Computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the data centers that provide those services. Cloud Computing has revolutionized the Information and Communication Technology (ICT) industry by enabling on-demand provisioning of elastic computing resources on a pay-as-you-go basis. An organization can either outsource its computational needs to the Cloud avoiding high up-front investments in a private computing infrastructure and consequent costs of maintenance and upgrades, or build a private Cloud data center to improve the resource management and provisioning processes.

Resource scheduling is a complicated task in a Cloud environment because of heterogeneity of the computing resources. To allocate the best resource to a Cloud workload is a tedious task and the problem of finding the best resource - workload pair according to Cloud consumer application requirements is an optimization problem. The main goal of the Cloud scheduler is to schedule the resources effectively and efficiently. Dispersion, heterogeneity and uncertainty of resources brings challenges to resource allocation, which cannot be satisfied with traditional resource allocation policies in Cloud circumstances.

In this thesis, the existing resource scheduling techniques have been discussed and compared. The different Cloud workloads and design patterns have been identified and analyzed. The classification of these workloads is done through K-Means Clustering Algorithm by assigning the weights to the different quality attributes. Resource scheduling is done on the basis of various scheduling criteria (Compromised Cost - Time Based, Cost Based, Time Based and Bargaining Based) selected by a decision tree. Resource scheduling algorithms for energy, time and cost constrained Cloud workloads have been proposed. The experimental results gathered through CloudSim 3.0 clearly demonstrate that the proposed framework has better performance for time, cost, energy etc. as compared to the existing scheduling algorithms.

Table of Contents

Certificate	i
Acknowledgement	ii
Abstract	iii
Table of Contents	iv
List of Figures	vi
List of Tables	viii
List of Abbreviations	ix
Chapter 1 Introduction	1
1.1 Cloud Computing Concepts	1
1.2 Cloud Computing Evolution	6
1.3 Cloud Architecture and Deployment Models	9
1.4 Open Issues in Cloud Computing	11
1.5 Research Motivation	13
1.6 Thesis Outline	15
Chapter 2 Literature Survey	16
2.1 Cloud Workloads	16
2.2 Resource Management in Cloud Computing	21
2.3 Cloud Design Pattern Based Approaches	30
2.4 Data Mining Algorithms	36
2.5 Conclusion	40
Chapter 3 Problem Analysis	41
3.1 Problem Statement	41
3.2 Objectives and Commitments	42
3.3 Conclusion	42
Chapter 4 Proposed Framework for Cloud Workloads	43
4.1 Cloud Workload Management Framework	43
4.2 Framework Assumptions	47
4.3 Framework Constraints	47
4.4 Framework Requirement Analysis	48
4.4.1 UML Diagrams	48
4.5 Framework Design	56
4.5.1 Design of CWMP Depicted Through DFD	56
4.6 Workload Identification and Analysis	58
4.7 Cloud Workload Based K-Means Algorithm for Clustering of Workloads	65
4.8 Conclusion	83

Chapter 5 Proposed Policies for Cloud Workloads	84
5.1 Resource Scheduling Procedure	84
5.2 Decision Tree Based Scheduling Criteria	85
5.3 Resource Scheduling Policies	92
5.3.1 FCFS Based Scheduling Policy	92
5.3.2 Compromised Cost - Time Based (CCTB) Scheduling Policy	92
5.3.3 Cost Based (CB) Scheduling Policy	96
5.3.4 Time Based (TB) Scheduling Policy	98
5.3.5 Bargaining Based (BB) Scheduling Policy	99
5.4 Conclusion	103
Chapter 6 Implementation and Experimental Results	104
6.1 Tools for Setting Cloud Environment	104
6.2 Framework Execution	108
6.3 Implementation of Proposed Framework	109
6.4 Results of Clustering of Cloud Workloads	120
6.5 Experimental Results	121
6.6 Conclusion	139
Chapter 7 Conclusions and Future Scope	140
7.1 Conclusions	140
7.2 Thesis Contribution	140
7.3 Future Directions	141
References	142
List of Publications	157

List of Figures

Figure 1.1:	Vision of Key Characteristics of Cloud Environment	1
Figure 1.2:	Sharing of Resources among Various Cloud Consumers	2
Figure 1.3:	Cloud Computing Elements and their Broad Classifications	5
Figure 1.4:	Emergence of Cloud Computing	7
Figure 1.5:	Cloud Computing Services	9
Figure 1.6:	Cloud Deployment Models	10
Figure 2.1:	Resource Distribution Policies	26
Figure 2.2:	Pattern Identification Process	33
Figure 2.3:	Precision-Recall Characteristics	39
Figure 2.4:	Comparison of Data Mining Algorithms	40
Figure 4.1:	Cloud Workload Management Framework	44
Figure 4.2:	Flowchart of Proposed Framework	46
Figure 4.3:	Use Case Diagram of the CWMP	49
Figure 4.4:	Sequence Diagram of the CWMP	50
Figure 4.5:	Activity Diagram of the CWMP	51
Figure 4.6:	Class Diagram of the CWMP	52
Figure 4.7:	State Chart Diagram of the CWMP	53
Figure 4.8:	Collaboration Diagram of the CWMP	54
Figure 4.9:	Component Diagram of the CWMP	54
Figure 4.10:	Modeling Executable Files and Libraries	55
Figure 4.11:	Deployment Diagram of CWMP	56
Figure 4.12:	Context or 0 Level DFD of the CWMP	57
Figure 4.13:	Level 1 DFD of the CWMP	58
Figure 4.14:	High Level Workload Groups	64
Figure 4.15:	K-Means Algorithm for Clustering of Cloud Workloads	69
Figure 4.16:	Distribution of Various Workloads among Four Clusters	73
Figure 4.17:	Design Pattern Template	75
Figure 4.18:	State Transition Diagram for Cloud Workload	78
Figure 4.19:	Cloud Resource Manager and its Associated Entities	79
Figure 4.20:	Sequence Diagram of Resource Monitoring	80
Figure 4.21:	Sequence Diagram of Workload Schedule	81
Figure 4.22:	Sequence Diagram of Dispatching Workloads	82
Figure 4.23:	Sequence Diagram of Workload Monitoring	82
Figure 4.24:	Class Diagram for Workload Allocation	83
Figure 5.1:	Resource Scheduling Procedure	84
Figure 5.2:	Decision Tree of Scheduling Policies	87
Figure 5.3:	Decision Tree Based Branching	89
Figure 5.4:	Compromised Cost - Time Based (CCTB) Scheduling Policy	94
Figure 5.5:	Cost Based (CB) Scheduling Policy	97
Figure 5.6:	Time Based (TB) Scheduling Policy	98
Figure 5.7:	Bargaining Based (BB) Scheduling Policy	102
Figure 6.1:	Multi-Layered Design of the CloudSim 3.0	106
Figure 6.2:	Integrate .NET and Java	107
Figure 6.3:	Implementation of IntegratedNETJavaWeb in ASP .NET	107
Figure 6.4:	Registration Form	109

Figure 6.5:	Registration Failed	110
Figure 6.6:	New User Details	110
Figure 6.7:	Successful Registration	111
Figure 6.8:	Login Page	111
Figure 6.9:	Wrong Username and/or Password	112
Figure 6.10:	Login Unsuccessful	112
Figure 6.11:	Correct Login Detail	113
Figure 6.12:	Workload Details	113
Figure 6.13:	Select Workload Name	114
Figure 6.14:	Select Preferred Policy	114
Figure 6.15:	Entered Workload Details	115
Figure 6.16:	Confirmation of Workload Details	115
Figure 6.17:	User Queries	116
Figure 6.18:	Cloud Provider Login	116
Figure 6.19:	Cloud Provider Home Page	117
Figure 6.20:	User Queries	117
Figure 6.21:	List of Registered Users	118
Figure 6.22:	Reply to User Queries	118
Figure 6.23:	Generating Workload Schedule	119
Figure 6.24:	Generated Workload Schedule	119
Figure 6.25:	Final Workload Schedule	120
Figure 6.26:	Attribute-Relation File Format File (k.means.arff)	120
Figure 6.27:	Scattering of Workloads amongst Four Clusters	121
Figure 6.28:	Execution of DBD-CTO in CloudSim	127
Figure 6.29:	Execution of CTC in CloudSim	127
Figure 6.30:	Execution of CCTB in CloudSim	128
Figure 6.31:	Execution Time Comparison of Cloud Workloads	128
Figure 6.32:	Cost Comparison of Cloud Workloads	129
Figure 6.33:	Execution Time for Different Number of Resources	129
Figure 6.34:	Cost for Different Number of Resources	130
Figure 6.35:	Execution Time for Different Number of Cloud Workloads	130
Figure 6.36:	Cost for Different Number of Cloud Workloads	131
Figure 6.37:	Number of Missed Deadlines	132
Figure 6.38:	Execution Time Variation	132
Figure 6.39:	Execution Cost Variation	133
Figure 6.40:	Energy Consumption and Number of Resources	134
Figure 6.41:	Completion Time vs. Allocated Budget	136
Figure 6.42:	Completion Time of Different Workloads	137
Figure 6.43:	Number of Deadlines Missed	138

List of Tables

Table 2.1:	Workload Description	17
Table 2.2:	Comparison of Existing Scheduling Algorithms	29
Table 2.3:	Comparison of Data Mining Algorithms	39
Table 4.1:	Workloads Based on Different Applications	63
Table 4.2:	Workloads Based on Resource Requirement and Programming Model	63
Table 4.3:	Classification of Cloud Workloads	64
Table 4.4:	Quality Attributes and Cloud Workloads Matrix	66
Table 4.5:	Questionnaire for Assigning the Weights to Quality Attributes	67
Table 4.6:	Conversion Metric	68
Table 4.7:	Workloads with their Requirements and Weights	70
Table 4.8:	Abbreviations of Workload Requirements	71
Table 4.9:	Data Values for Workloads	71
Table 4.10:	The Four Seeds for Given Workloads	71
Table 4.11:	First Iteration- Allocating Each Cloud Workload to the Nearest Cluster	72
Table 4.12:	Comparing New Centroids and the Seeds	72
Table 4.13:	Second Iteration- Allocating Each Cloud Workload to the Nearest Cluster	73
Table 4.14:	Cluster Membership	73
Table 4.15:	Clusters with Patterns and Cloud Workloads	76
Table 5.1:	Policies Description	86
Table 5.2:	Compromised Cost - Time Based Rules	90
Table 5.3:	Time Based Rules	91
Table 5.4:	Cost Based Rules	91
Table 5.5:	Bargaining Based Rules	92
Table 6.1:	Cloud Workload Details	122
Table 6.2:	Scheduling Parameters and their Values	122
Table 6.3:	Cost Related to Different Processing Speed	123
Table 6.4:	Deadline Urgency	124
Table 6.5:	Total Expected Cost and Total Expected Completion Time	124
Table 6.6:	Time Difference	125
Table 6.7:	Rescheduling of Cloud Workloads	125
Table 6.8:	Actual and Improved Execution time	135
Table 6.9:	Cloud Workloads Detail	137
Table 6.10:	Resource Detail	138

List of Abbreviations

Notation	Definition
UNIVAC	UNIVersal Automatic Computer
ERP	Enterprise Resourcing Planning
OS	Operating System
MPP	Massively Parallel Processing
SMP	Symmetric Multi-Processing
UPS	Uninterrupted Power Supply
PC	Personal Computer
CPU	Central Processing Unit
ASP	Active Server Pages
SLA	Service Level Agreement
LAN	Local Area Network
QoS	Quality of Service
HICCAM	Hybrid Cloud Construction And Management
SHEFT	Scalable Heterogeneous Earliest Finish Time
RASA	Resource Aware Scheduling Algorithm
SwinDeW-C	Swinburne Decentralized Workflow for Cloud
GUI	Graphical User Interface
JVM	Java Virtual Machine
JAR	Java Archive (JAR) File Format
CCTB	Compromised Cost - Time Based (CCTB) Scheduling Policy
CTC	Compromised Time Cost (CTC) Scheduling Policy
DBD-CTO	Deadline and Budget Distribution-Based Cost-Time Optimization (DBD-CTO) Workflow Scheduling Policy
AWS	Amazon Web Services
SPEC	Standard Performance Evaluation Corporation
MIPS	Million Instructions Per Second
IT	Information Technology
SAP	Systems Applications and Products
PE	Processing Element

Chapter 1

Introduction

This chapter provides an overview of Cloud Computing, Cloud Computing evolution, deployment models and architecture, elements of Cloud Computing and many open research issues. It briefly presents the research motivation for Cloud Computing and presents primary contributions of this research. In the last, the chapter discusses the organization of the rest of this thesis.

1.1 Cloud Computing Concepts

Cloud Computing is a model for permitting omnipresent, suitable, on-demand service access to a common group of configurable Computing resources (e.g., networks, servers, storage and applications) that can be quickly provided and released with least management struggle or Cloud provider dealings [1]. The vision of key characteristics of Cloud environment is shown in Figure 1.1.

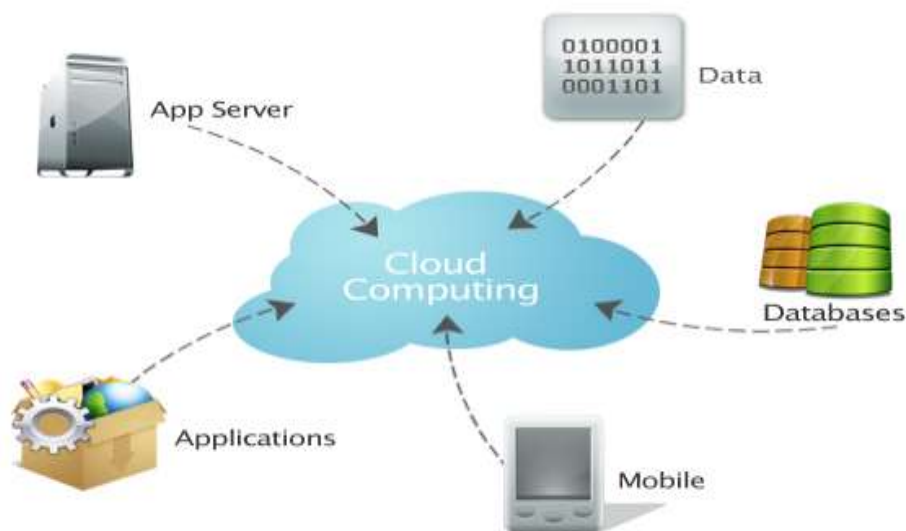


Figure 1.1: Vision of Key Characteristics of Cloud Environment [2]

Public Cloud platforms are usually superior at providing IT services over the open Internet than enterprise IT is proficient of doing. Therefore, the public Cloud can well

serve a workforce that's expected to work at the local region because processing, storage, and enterprise applications to a middle tier between the company and the Cloud consumer can drive easily. Any company with a mobile or geographically distinct workforce could profit from Cloud storage and other third-party services as shown in Figure 1.2 that shows the sharing of resources among various Cloud consumers. The painting is from Computer History Museum, Mountain View, California, USA [3]. Through Cloud Computing, the organizations improve efficiency, sustain innovation and improve motivation. Dealing with fluctuations in requirement much more economically, the public Cloud service provider's stream distributes the workload over various consumers and passes the investments from our economies of scaling to you.



Figure 1.2: Sharing of Resources among Various Cloud Consumers [3]

Cloud Computing comprises communal digital resources that are retrieved via a credit card application programming customer interface. The authentic resources might be distributed geographically, inside or outside the organization. To obtain these resources from the faraway company, a payment is provided according to the usage of decent enterprise information center. With the actual Cloud, users can certainly access extra resources when user wants them. The way of consuming Cloud processing and available resources will be automated; its resources will be an offer to any or all Cloud consumers every time in all places by flexible management of resources. Cloud Computing is the access to systems and their functionality via the Internet or a LAN. Cloud consumers of a

Cloud require this access from a set of web services that preserve a group of Computing resources (OS, network, system, storage).

The different features of Cloud Computing [4] are summarized below:

- **Economical:** Cloud Computing is economical because of usage based billing model, no need of infrastructure.
- **Larger Storage Space:** With the huge Infrastructure, storage & management of large space is possible. Sudden Cloud workload fluctuations are also managed successfully, since the Cloud can scale dynamically during the situation of overloading and under loading.
- **Elasticity:** Cloud Computing stresses on receiving Cloud workloads or applications to market very rapidly, by using the most suitable building blocks essential for deployment.
- **Trustworthiness:** It's ability to confirm continuous working of computers without interruption, i.e. no damage of data, no code change throughout execution etc.
- **Geographically Distributed:** ¹Cloud consumers can use resources through a web browser irrespective of their site from where user are accessing.
- **Dynamic Scalability:** The data and application resources can be rapidly provided when and where wanted.
- **Availability:** With the right Cloud provider, it will be guaranteed that available resources remain continuously accessible i.e. 24 X 7.
- **Less Management:** The hardware, applications and bandwidth are maintained by the Cloud provider.
- **Expert Service:** At convenience, Cloud Computing facilities are continuously supervised and maintained by onsite staff of professional data center specialists.

1.1.1 Business Advantages

The following are main business advantages to develop applications through Cloud Computing [5] [6] [7]:

¹ The terms “user”, “Cloud customer” and “Cloud consumer” are synonymous and interchangeable.

- **No Direct Infrastructure Expenditure:** When a large-scale system will be developed it includes the cost of hardware (frames, systems, routers, UPS), hardware maintenance (power management, cooling), and staff management. Because of the direct costs, it would normally require some stages of supervision appreciations before the project starts. Currently, there is no fixed charge with the invention of utility computing.
- **JIT Infrastructure:** Through executing applications in the Cloud with dynamic capacity management software architects do not have to worry about advance acquiring capacity for large-scale systems. The solutions are little threat because scalability depends on Cloud user project size. Cloud architectures can abandon infrastructure as rapidly as Cloud user will get within minutes.
- **Maximum Resource Utilization:** Through Cloud architectures Cloud provider can manage resources more efficiently by taking the applications request and release free resources.
- **Consumption Based Charges:** Cloud Computing based on utility permits billing the Cloud consumer only for the resources that will be acquired. The Cloud consumer is not responsible for the whole infrastructure that might be in place.
- **Reduce Execution Time:** The processing speed will be increased through the concept of parallelization. If one computes intensive Cloud workload that can be executed in parallel it takes 500 hours to execute on a single system. With Cloud architectures, it would be possible to spawn and launch 500 instances and execute similar Cloud workload in 1 hour. An elastic infrastructure delivers the application with the capability to exploit parallelization in an economical style decreasing the total execution time.

1.1.2 Elements of Cloud Computing

There are seven main elements of Cloud Computing, classified on economic, architectural and strategic elements' [8] basis as shown in Figure 1.3.

- **Utility Pricing -** Cloud Computing is well-defined by its usage based billing model. Cloud provider contributes to this as Cloud consumers of the platform use

computing and storage facilities on requirement and pay based on the consumption, using an Operating Expenses (OPEX) budget and Capital Expenditures (CAPEX) [9].

- Elastic Resource Capacity - Cloud Computing scales computing and storage resources up and down, consumers can add or remove resources immediately and make payment for the resources a Cloud consumer is consuming [10].
- Virtualized Resources - Without virtualization Cloud Computing is impossible, not for mysterious technical causes, but for one recognizable business requirement: the requirement of multi-tenancy. In order to take advantage from saving of scale, Cloud Computing is based upon the distribution of a shared infrastructure by many sets of Cloud consumers, frequently denoted to as renters [5].

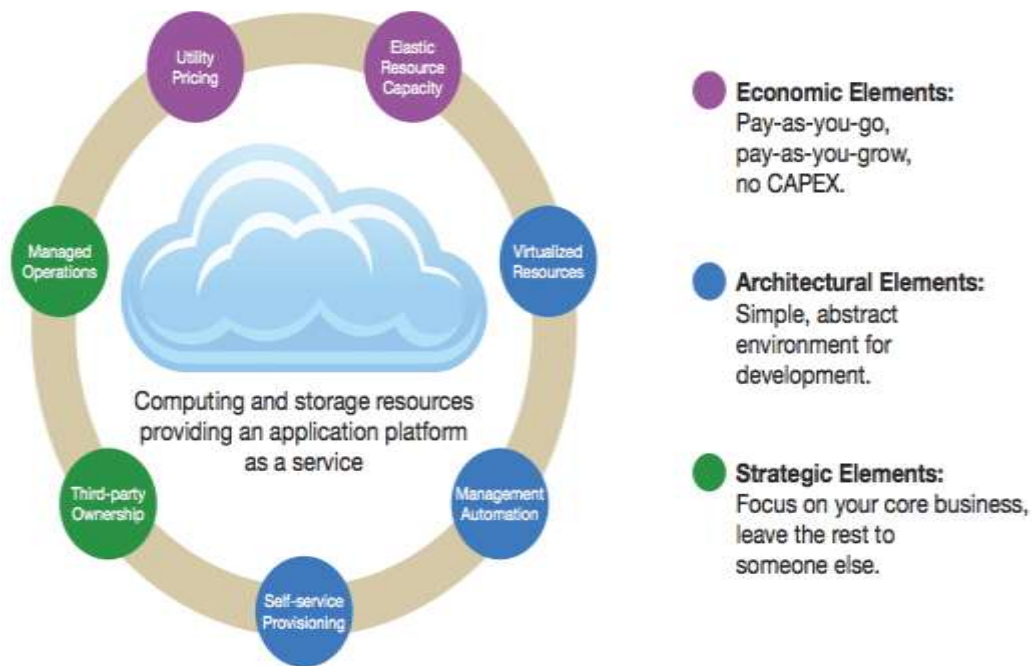


Figure 1.3: Cloud Computing Elements and their Broad Classifications [11]

- Management Automation - Standardization makes Cloud Computing different from traditional corporate data-centers by dramatically reducing operational costs through aggressive management automation. Though usual data-center, provider will typically host each varieties of each OS and databases recognized to manhood,

thus generating huge management overhead, most Cloud Computing platforms generally standardize on a single kind of CPU (x86-based primarily), a single hypervisor (VMware), a single OS (Linux distribution commonly), and a single database (MySQL rules) [12].

- Self Service Provisioning - Application Service Provider (ASP) model that became famous for short time is compared with Cloud Computing and Software as a Service in context of self service provisioning. Cloud consumer like Director of Sales can deliver applications and customer accounts in a few mouse clicks, and these become accessible immediately with Cloud Computing [5].
- Third Party Ownership - Users demanding to emphasize the distribution of occasional principal resources to their primary businesses quickly recognize the reimbursements of moving IT infrastructure off their balance sheet. Moreover, as technology changes and prominent Cloud providers roll-out ever larger data-centers, the procurement and operation of state of the art data-center services makes less and less sense from a commercial perspective for most companies. Cloud Computing is all about the handover ownership for such resources to a third-party that concentrates in their deployment [8].
- Managed Operations - Assigning human resources to tasks that will openly affect the business, rather than handling the infrastructure in Cloud Computing, this advocates a model according to the IT infrastructure which is retained and maintained by the third party. Maintenance of software's, data replication and countless other tasks is mandatory to handle mission-critical business applications on a daily basis that becomes the accountability of a third party, according to SLAs [5].

1.2 Cloud Computing Evolution

Cloud Computing can be realized as a revolution in many ways. From a scientific viewpoint it is an improvement of computing, relating virtualization notions to use hardware more proficiently. In this sense Cloud Computing has the power to modernize the approach, how computing resources and applications are delivered, breaking up old significance chains and creating an opportunity for innovative commercial models [13]

[14]. Firstly, limited public beta of EC2 (Amazon Elastic Compute Cloud) was provided by Amazon in 2006. At the time no one could have recognized how popular and transformative virtual resources over the Internet would become [5]. This section explains the emergence of Cloud Computing from proprietary mainframe as shown in Figure 1.4.

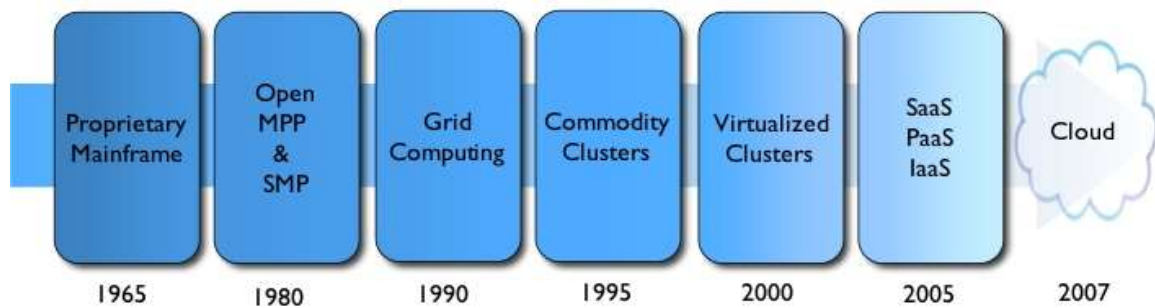


Figure 1.4: Emergence of Cloud Computing [14]

- **Proprietary Mainframes**
Back in the 1950s to early 1970s, “IBM and the Seven Dwarfs” presented the mainframe computer. This group included IBM, UNIVAC, Control Data and General Electric. Almost all mainframes had the capability to execute many OSs, and in this manner operate not as a single computer but as a number of Virtual Machines. The mainframe serviced critical applications (usually bulk data processing) including ERP and commercial transaction processing [15].
- **Open MPP & SMP**
Reducing demand and hard competition caused a revolution in the market in the early 1980s. Corporations found that servers based on parallel microcomputer designs (primarily classified as either MPP or SMP) might be deployed at a fraction of the achievement charge and suggest local Cloud consumers much better control over their own systems. Terminals used for interrelating with mainframe systems were progressively substituted by personal computers network linked to servers [15].
- **Grid Computing**
Grid Computing is the greatest distributed form of Parallel Computing. It enables usage of computers interconnecting over the Internet to work on a specified

problem. For that reason, low bandwidth and tremendously high latency available on the Internet, Grid Computing normally deals only with awkwardly parallel problems. The trend of Grid happened in the 1990s with an approach to resolve enormous problems such as financial modeling, weather demonstrating and earthquake simulation [16].

- Commodity Clusters

In 1990, the Cluster Computing originated. Clusters are loosely coupled “commodity” servers typically deployed to increase performance or availability over that of a single system (e.g. Proprietary Mainframe, MPP, or SMP server), and as well considerably more economical than single systems of comparable speed or availability [17].

- Virtualized Clusters

A Virtual Machine (VM) is a software implementation of a machine (server) that performs programs similar to an actual system. The hardware VMs (System VMs) permit the sharing of the fundamental actual system resources among different VMs, each executing its own OS. The software layer providing the virtualization is called a VM hypervisor. A hypervisor can execute on bare hardware (native VM) or on top of an OS (hosted VM) [13].

- Cloud Services (IaaS, PaaS, SaaS)

Once the hardware upon which applications were developed expressively reduced in worth, corporations initiated pooling resources and permitting small to average sized businesses access to compute resources based on requirement. These corporations established services accessed through web browser denoted to as IaaS (Infrastructure as a Service), or PaaS (Platform as a Service), or SaaS (Software as a Service), generally it is denoted by – XaaS where $X = \{I, P, S, \dots\}$ and after sometime it became generally known as Cloud [1] [13]. These three main types of services provided by Cloud Computing are shown in Figure 1.5.

- Infrastructure as a Service (IaaS): Through IaaS the delivery of resources to the Cloud consumers such as servers, storage, and associated tools essential over the Internet, permitting enterprises to develop an application environment from scratch based on requirement is very easy

and inexpensive. Billing is based on the usage of service and can get complicated with tiered on-demand valuing. Amazon is an IaaS provider.

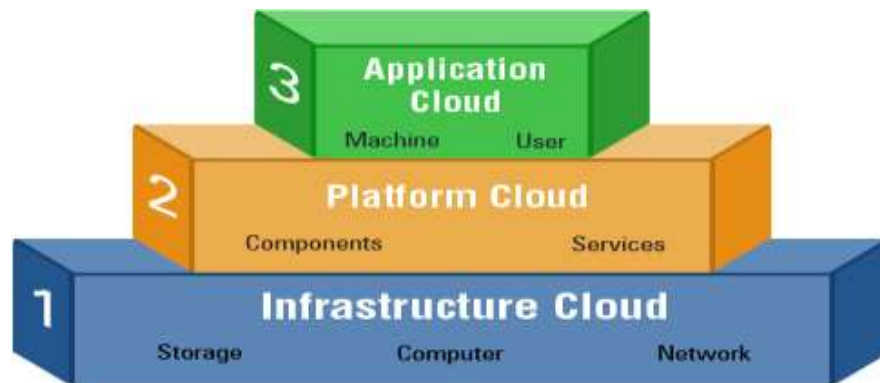


Figure 1.5: Cloud Computing Services [18]

- Platform as a Service (PaaS): PaaS enables the deployment and scalability of user application trivial and charges incremental and practicably foreseeable –a user can deploy applications on PaaS. Google and Microsoft is PaaS provider [13]. User can execute their application with little modification and probably execute already existed applications with efforts.
- Software as a Service (SaaS): SaaS or application Cloud is one type of Cloud services, where software functionality is delivered as a service. SaaS provides benefits to service consumers; no initial cost to purchase software, free of maintenance/updates, accessible through the Internet, high availability, and pay-per-use pricing. SaaS should preserve a comparatively greater level of its quality than conventional system. The most renowned vendors and useful examples of SaaS are GMail or Salesforce [13].

1.3 Cloud Architecture and Deployment Models

Cloud architectures are designs of software applications that use Internet-accessible on-demand services. Applications developed on Cloud architectures are such that the fundamental computing infrastructure is used only when it is required (to process a Cloud consumer request), draw the essential resources on-demand (compute servers or storage),

perform a particular job, then hand over the unnecessary resources and frequently organize themselves after the job is completed. Though in process the application scales up or down elastically based on resource requirements [19]. Cloud architecture addresses important problems adjacent to large-scale data processing. It is not easy to do automatic scalability based on vibrant Cloud workloads and it is difficult to become free of all those systems when the task is completed. Cloud architectures resolve such problems.

Cloud Computing can be run in four deployment models as shown in Figure 1.6. The deployment model will be used depending on the Cloud consumer desire and on market availability [6].

- Private Cloud - A private Cloud can be executed internally or by a Cloud provider (third-party) benefits of which cannot be fully exploited, and the degree of customization probably might be inadequate.
- Community Cloud - The service is used by some members of a well-defined group and it is provided by several Cloud providers who are either internal or external to the community.
- Public Cloud - The service is offered to the public, and in general delivered by a single Cloud provider in which scalability and resource pooling can be completely exploited.

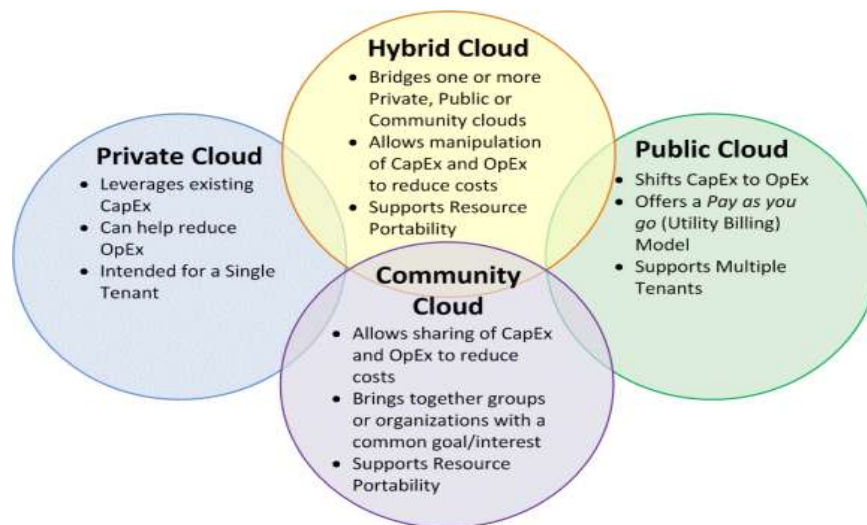


Figure 1.6: Cloud Deployment Models [20]

- Hybrid Cloud - Hybrid Clouds provide a grouping of many organization procedures, combining their respective benefits and drawbacks. For example, data that need to be secure can reside in a private Cloud, whereas public data or applications can execute in the public Cloud.

1.4 Open Issues in Cloud Computing

The beginning of Cloud Computing has made an incredible influence on the IT industry over the past few years. Presently IT industry wants Cloud Computing services to offer best chances to real world. Cloud Computing is in early stages, with many problems [21] [22] yet to be addressed. Some of the open issues of Cloud Computing are:

i. Automatic Service Providing

The aim of a Cloud provider in this case is to assign and release resources from the Cloud to fulfill its SLOs (Service Level Objectives), reducing its deployment charge. These methods usually include: (i) Creating an application performance model that forecasts the number of application instances needed to manage request at every individual level, in order to fulfill QoS requirements; (ii) Occasionally forecasting forthcoming demand and defining resource requirements using the performance model; and (iii) Automatically assigning resources using the forecast resource requirements. The proactive method uses forecast demand to occasionally assign resources before resources are required. The reactive method responds to instant demand variations before periodic demand forecast is accessible [21].

ii. Virtual Machine Migration and Server Consolidation

Virtualization can deliver important profits by allowing Virtual Machine migration to stable workload across the data center. Further, VM migration permits strong and highly responsive providing in data centers. Researchers have found that moving a whole OS and all of its applications as one unit allows avoiding many of the problems tackled by process-level migration methods, and investigated the advantages of migration of VMs. Detecting workload hotspots and initiating a migration lacks the agility to respond to sudden workload changes [22]. Server consolidation is an operative method to improve resource utilization by reducing energy consumption.

Energy can be saved through VM migration. To combine VMs, VMs should be located on many under-utilized servers onto a single server [21].

iii. Energy Management

In Cloud Computing, the improvement in energy efficiency is one of the major problems. It has been assessed that the price of powering and refrigeration accounts for 53% of the entire operational spending of data centers [23]. In 2006, data centers in the US consumed more than 1.5% of the total energy produced in that year, and the proportion is estimated to grow 18% yearly [23]. Therefore infrastructure providers are under huge pressure to decrease energy consumption. The aim is not only to decrease energy cost in data centers, but also to meet government rules and environmental standards. Energy-oriented task scheduling and server consolidation are two other methods to decrease power consumption by switching off free systems. A main issue in all the above techniques is to attain a decent trade-off between energy savings and application performance [21].

iv. Data Security

Data security is another open issue in Cloud Computing. Meanwhile Cloud providers usually do not have access to the physical data security system of data centers; Cloud provider must depend on the infrastructure provider to attain complete data security. Even for a virtual private Cloud, the Cloud provider can only identify the security setting distantly, without knowing whether it is completely implemented or not. It is dangerous to form trust procedures at each architectural layer of the Cloud. Initially, the hardware layer must be reliable using hardware reliable platform segment. Furthermore, the virtualization platform need be confidential using secure Virtual Machine observers. VM migration should only be permitted if both sender and receiver servers are confidential [22].

v. Resource Scheduling and Dynamic Scalability

The aim of scaling and resource scheduling is to maximize application performance within budget constraints in Cloud workloads [21]. What resources should be acquired/released in the Cloud, and how should the computing activities be mapped to the Cloud resources, so that the application performance can be maximized within the budget constrains? Dynamic scalability is the ability to acquire or release the

resources in response to demand dynamically. In a data center, the primary goal of a dynamic autonomous resource management process is to avoid wasting resources as a result of under-utilization [22]. Such a process should also aim to avoid high response times as a result of over-utilization which may result in violation of the Service Level Agreement (SLA) between the clients and the provider [21]. To design a successful IaaS, initially understand the Cloud workload (e.g. transactional database, file server, web server, application server and batch data processing) thoroughly. Based on this, Cloud consumer should design their applications which to lead to maximization of the scaling advantage. With the help of this, not only dynamic infrastructure scaling can be achieved but it will minimize the response time of elastic demand and maximize the throughput of requests [19]. It is difficult to prepare an IT resource to fulfill its processing desires. IT resources may be over-utilized or under-utilized depending on demand.

1.5 Research Motivation

Resource Scheduling is the practice of implementing policies and procedures that improve the efficiency of computing resources in such a way so as to reduce the execution time and cost, energy consumption and environmental impact of their execution. There are many issues in Cloud Computing such as Automated Service Provisioning, Virtual Machine Migration, Energy Management, Server Consolidation, Data Security, Dynamic Scalability and Resource Scheduling etc. as discussed in previous section that have not been fully addressed [20] [21].

In Cloud Computing environments, there are two parties: Cloud providers and Cloud users. On one hand, providers hold massive computing resources in their large datacenters and rent resources out to users on a per-usage basis. On the other hand, there are users who have applications with fluctuating loads and lease resources from providers to run their applications. One interesting aspect of the Cloud Computing environment is that these parties are often different parties with their own interests. Typically, the goal of providers is to generate as much revenue as possible with minimum investment. To that end, they might want to squeeze their computing resources; for example, by hosting as many workloads as possible on each resource. In other words, providers want to

maximize resource utilization. However, placing too many workloads on a single resource will cause workloads to interfere with each other and result in degraded and/or unpredictable performance which, in turn, frustrates the users. Thus, the providers may evict existing resources or reject resource requests to maintain service quality, but it could make the environment even more unpredictable. On the other hand, users want their workloads done at minimal expense or, in other words, they seek to maximize their cost performance. This involves having proper resources that suit the workload characteristics of users' applications and utilize resources effectively. They also have to take unpredictable resources into account when they request resources and schedule applications. However, these two parties do not want to share information with each other, which makes optimal resource allocation more difficult [20].

Dispersion, heterogeneity and uncertainty of resources brings challenges to resource allocation, which cannot be satisfied with traditional resource allocation policies in Cloud circumstances. Thus, there is a need to make Cloud services and Cloud-oriented applications efficient by taking care of these properties of the Cloud environment. Aim of resource scheduling is to allocate appropriate resources at the right time to the right workloads, so that applications can utilize the resources effectively [21]. In other words, the amount of resources should be minimum for a workload to maintain a desirable level of service quality, or maximize throughput (or minimize workload completion time) of a workload. To address this problem, new solution should be developed. The work in this thesis aims to support the scheduling of multiple workloads, to be able to meet multiple QoS requirements for Cloud workload. Therefore, a Cloud workload management framework has been presented which would not only consider the completion time of each single workload, but most importantly, the overall performance also. The overall performance can be defined in various aspects. Among them, this thesis focuses on the following aspects: (1) the energy consumption; (2) the execution time of all workloads - the algorithms must strive to decrease the execution time of all workloads; and (3) the execution cost of all workloads - the algorithms should minimize the cost of workload on the basis of meeting the QoS requirements of time.

1.6 Thesis Outline

Rest of chapters in this thesis is organized as:

Chapter 2 – This chapter provides various Cloud workloads in Cloud Computing, Resource Scheduling Policies, Cloud Design Pattern Based Approaches and Data Mining Algorithms.

Chapter 3 – This chapter discusses the Problem Statement and Objectives and Commitment.

Chapter 4 – This chapter describes Cloud Workload Management Framework, Framework Requirement Analysis, Framework Design, Workload Identification and Analysis and Cloud Workload based K-Means Algorithm for Clustering of Workloads.

Chapter 5 - This chapter discusses Resource Scheduling Procedure, Decision Tree based Scheduling Criteria and Resource Scheduling Policies.

Chapter 6 – This chapter describes the Tools for Setting Cloud Environment, Framework Execution, Implementation of Proposed Framework, Clustering of Cloud Workloads and Experimental Results.

Chapter 7 – This chapter summarizes the Conclusions drawn in the thesis along with Thesis Contribution and Future Research Directions.

This chapter provides related work of the various Cloud workloads in Cloud Computing, resource scheduling policies, Cloud design pattern based approaches and data mining algorithms.

2.1 Cloud Workloads

Cloud workload is an abstraction of work of that instance or set of instances going to perform [24]. For Example: Running a web services or being a Hadoop data node are valid workloads. A workload is a self-governing services or group of code that can be implemented; workload does not depend on outside demands [27].

2.1.1 Workload Taxonomy

The server workload classification and guess have been considered mostly in the past. Nevertheless, most existing works have a slightly different emphasis or use a different method [26]. Different from these works, which emphasis on workload modelling at distinct server or general data centre level, there study emphases on knowing the interrelated workload patterns within clusters of servers that result from application dependencies [26]. Moreover linked to this analysis are the works on capacity management and virtual server placement, which usually employ some workload modelling and forecast methods [30]. Different researchers' work on Cloud workload has been summarized in Table 2.1. These methods answer on the statistics (e.g., percentiles, peaks etc.) of distinct workload time series to forecast upcoming capacity demand [30]. In comparison, some other methods discover inter-server relationships and try to forecast workload levels at finer granularity. Arijit Khan et al. [31] described data traces acquired from a real data centre to progress such abilities. Initially, they have searched for repeatable workload patterns by discovering cross-server performance relationships resulted from the dependencies among applications running on dissimilar servers. Considering server workload data models as multiple time series, they suggest a co-clustering technique to identify groups of servers that frequently show associated workload patterns, and also the time intervals in which these server groups are energetic.

Table 2.1: Workload Description

Authors	Description
Walfredo Cirne et al. [24]	Established hypothetical models to create illustrative workload traces.
Arlitt et al. [25]	Analysed the workload classification on Web servers.
Cherkasova et al. [26]	Conducted an analysis on broadcasting servers.
Gmach et al. [27]	Considered the workload for data centre applications.
Bobroff et al. [28]	Used regression models to prediction workload deviations, in order to dynamically place Virtual Machines.
Verma et al. [29]	Suggested consolidating servers using association or peak cluster based assignment.
Jerry Rolia [30]	A trace-based workload predicting technique was used for capacity management.

They also presented a technique based on Hidden Markov Modelling (HMM) to classify the time-based correlations in the exposed server clusters and to forecast variations in workload patterns [32]. The experimental outcomes of suggested approach describe that method can not only support Cloud providers better recognize group-level workload characteristics in a Cloud, but also make extra accurate forecasts on workload as it varies over time. They used Hidden Markov Model (HMM) to explore the temporal correlations in workload pattern changes to forecast distinct server performance based on the server groups found in the earlier stage [32].

Based on actual measurement data gathered from production Cloud environment, therefore offers visions for Cloud providers to know the distinctive Cloud workload patterns and to better manage resources. To recognize common CPU utilization patterns across multiple servers by workload classification [25]. To attain that discretize workload time series into some discrete levels, then search for collections of servers that often display definite groupings of workload levels together [27]. But this technique does not suppose availability of whole information about application dependencies among servers, but determines and controls such dependencies by probing for repeatable workload patterns within set of servers [28]. The co-clustering method is used to recognizing such server groups and the common workload patterns. The recognized co-clusters offer a significant worth headed for understanding Cloud workload features. Moreover, a HMM-based method developed to capture the time-

based relationships in workload changes across different co-clusters to forecast workload variation at a cluster level [29].

2.1.2 Workload Analysis on Virtual Environment

Shiang-Jiun Chen et al. [33] inspected the performance aspects of virtualization standard based on the VMmark model which adapts the identical workloads on single server. In real circumstances, several workloads would execute on the identical server, the consequences of new scores may be different with the original metrics. They proposed an experiment like VMmark benchmarking to assess the workloads of virtual systems.

2.1.3 Cost and Time Based Optimized Workload Scheduling

The aim of this judgment is to maximize the utilization of the internal data center and to diminish the cost of running the outsourced jobs in the Cloud, even though satisfying the applications' QoS constraints. Ruben Van den Bossche et al. [34] studied this optimization problem in a multi-provider hybrid Cloud setting with deadline-constrained and pre-emptible but non-provider migrateable workloads that are categorized by memory, CPU and data transmission desires. Linear programming is a common practice to handle such an optimization problem. At up-to-date, it is still uncertain whether this practice is right for this problem and what the performance allegations of its use are.

2.1.4 Cost-effective and Durable Provisioning of Cloud Workloads

There are two main causes for provisioning of resources for N-tier web applications in Clouds: 1) An essential optimization struggle between cost of resources and Service Level Agreement (SLA) compliance and 2) The resource demands of the multiple tiers are not similar, and changing along with the time. Resources have to be assigned to multiple (virtual) containers to reduce the total amount of resources while fulfilling the end-to-end performance desires for the application. Pengcheng Xiong et al. [35] addressed these two open issues through the grouping of the resource controllers on both application and container levels. A decision maker controls the total cost of the resources that are mandatory for the application to meet SLA desires as the workload fluctuates at the application level. On the other hand, supervisor divides the entire

resource budget among the components of the applications to optimize the application performance at container level (i.e., to reduce the round trip time).

2.1.5 Execution of Cloud Workloads

Konstantinos Tsakalozos et al. [36] suggested Nefeli, a virtual infrastructure gateway that is able of efficiently control different Cloud workloads. Cloud customers and their workloads remain doubtful to the internal features of Clouds at all times. Misusing execution patterns as well as logistical constraints, consumers offer Nefeli with hints for the handling of their Cloud workloads. Nefeli benefits avoid bottlenecks within the Cloud through the recognition of feasible Virtual Machine deployment mappings. As the types of Cloud workloads change over time, deployment mappings must follow uniformity. To this end, Nefeli provides methods to migrate Virtual Machines as required adjusting to varying performance requirements.

2.1.6 Online Cost-Efficient Scheduling of Deadline-Constrained Workloads

The use of hybrid Clouds presents the requirement to define which Cloud workloads are to be outsourced, and to what Cloud provider. These judgments must reduce the cost of executing a partition of the total workload on one or multiple public Cloud providers while considering application desires such as data requirements and deadline constraints. Ruben Van den Bossche et al. [37], handled this limitation by suggesting a set of procedures to cost efficiently schedule deadline-constrained bag-of-tasks applications on both public Cloud providers and private infrastructure. This approach is better taken care of both computational and data transfer costs as well as network bandwidth constraints. The performance has been evaluated in a realistic environment with respect to fulfil deadlines and computational effectiveness, economical and examines the influence of faults in runtime evaluations on these performance metrics.

2.1.7 Workload Analysis for Power Minimization

Server consolidation has emerged as an encouraging method to reduce the energy costs of a data center. Akshat Verma et al. [29] presented the first thorough investigation of an enterprise server workload from the viewpoint of finding features for consolidation. The significant potential for power savings if consolidation is achieved using off-peak values for application demand has been discussed. Though,

these savings come up with related risks due to consolidation, predominantly when the correlation between applications is not deliberated. The steadiness in utilization trends for low-risk consolidation has been investigated. Using the visions from the workload study, two new consolidation approaches are considered save power, but holding the consolidation performance risk.

2.1.8 Effects of Workload on Virtual Machine Performance

George Kousiouris et al. [38] predicted the influence of a number of critical factors on the performance of Virtual Machines (VMs). These factors contain allocation percentages, real-time scheduling judgments and co-placement of VMs when these are deployed simultaneously on the similar physical node. Based on a multi-core architecture (coupling of VMs to cores), the different levels of memory sharing are used.

2.1.9 Capacity Planning and Performance Management through Workloads

Efficient classification of workload could be used to drive Capacity Planning and Performance Management in IaaS Cloud. There are different workload metrics (e.g. CPU, memory usage, throughput, response time) which could be demonstrated along with relations among them. Likewise, model relationships across a set of workloads have been identified. Examining and characterising this would allow decision making for several situations such as VM migration, resource management and re-provisioning. Shruti Mahambre et al. [39] studied workload executing in IaaS Cloud and classify into patterns, based on their behavioural features. The different categories of behavioural patterns and outline statistical methods to be used in defining these patterns have been defined.

2.1.10 Workload Factoring

Cloud Computing is an evolving paradigm in which workloads are allocated to an aggregation (“Cloud”) of servers and devices, retrieved over a network. Usually, the Cloud establishes an extra means of computation and a Cloud customer can perform workload factoring, i.e., fragment its load among the Cloud and its other resources. Based on empirical data, there is a basic relation between the “benefit” that a Cloud consumer perceives from the Cloud and the usage pattern followed by other Cloud customers [40].

2.2 Resource Management in Cloud Computing

Cloud Computing is on demand as it offers dynamic flexible resource allocation for reliable and guaranteed services in pay according to use. Many Cloud consumers can demand number of Cloud services concurrently in Cloud Computing. Subsequently there is a need to provide all the resources to requesting Cloud consumer in a well-organized way to fulfil their requirements. There are different ways to allocate the resources to Cloud workloads have identified from the literature as given below:

2.2.1 Workflow Scheduling

Workflow scheduling is the problem of mapping each workload to suitable resource and permitting the workloads to fulfil some performance standard. A workflow contains of a sequence of concatenated (connected) stages. Workflow is generally concentrated with the automation of processes and also in order to attain an overall objective, files and data are passed between participants according to a defined set of instructions. A workflow permits the organizing of applications in a Directed Acyclic Graph form where each node signifies the workload and edges represent the dependencies among the workloads of the applications. A particular workflow consists of a set of workloads and each workload communicates with another workload in the workflow. Workflows are supported by Workflow Management Systems. Workflow scheduling determines resources and allocates workloads on suitable resources. Workflow scheduling plays a vibrant role in the workflow management. Appropriate scheduling of workflow can have an effective influence on the performance of the system. For appropriate scheduling in workflows several scheduling algorithms are used [41].

Suraj Pandey et al. [41] presented a Particle Swarm Optimization (PSO) based heuristic to schedule the applications to Cloud resources that proceeds both computation and data transmission cost. It is used for workflow applications by changing its computation and communication costs. The assessment results show that PSO can reduce the cost and good sharing of workload onto resources. Meng Xu et al. [42] worked on many workflows and multiple QoS. They executed an approach for multiple workflow management system with multiple QoS. The access speed for scheduling is improved by using this approach. This approach decreases the makespan

and cost of workflows. Topcuoglu et al. [43] presented the HEFT algorithm to discover the average execution time of each workload and also the average communication time among the resources of two workloads. Then workloads in the workflow are well-ordered on a rank function. Then the workload with higher rank value is given higher priority. In the resource selection stage workloads are scheduled in priorities and each workload is allocated to the resource that complete the workload at the earliest time. Zhangjun Wu et al. [44] suggested a market oriented hierarchical scheduling approach which contains of both service level scheduling and workload level scheduling. The service level scheduling deals with the Task to Service assignment and the workload level scheduling deals with the optimization of the Task to Virtual Machine assignment in local Cloud data centers. Jia Yu et al. [45] proposed a cost based workflow scheduling algorithm reduces the execution cost however meeting the deadline for delivering results. It can also adjust to the delays of service accomplishments by rescheduling unexecuted workloads. Rizos Sakellariou et al. [46] suggested a simple model for workflow applications that modelled as Directed Acyclic Graph (DAG) and that permit to schedule the nodes of DAG onto resources in a method that fulfils a budget constraint and is optimized for overall time.

2.2.2 Traditional Scheduling Algorithms

Saeed Parsa et al. [47] suggested a different task scheduling algorithm i.e. RASA. It is aggregation of Max-min and Min-min scheduling algorithms. They suggested that the incoming rate of jobs, deadline of every task, cost of the job processing on every of the resource and communication cost are considered. For large scale distributed systems RASA is better than traditional scheduling algorithms. Arash Ghorbannia et al. [48] proposed a trustworthy scheduling algorithm in Cloud Computing. The main problem is divided into sub problems (Jobs). The request and acknowledge time are computed independently in the form of a shared job to make it balanced. M. Dakshayini et al. [49] offered a priority based scheduling algorithm. The priority is given to every submitted queue. Submission of every queue is decided by calculating acceptable delay and service charge. By this algorithm Cloud architecture has attained great (99%) service completion speed with assured Quality of Service but whole servicing cost for the Cloud also rises. Shamsollah Ghanbari et al. [50] proposed a multi-criteria and multi-decision priority driven based scheduling. Object level, attribute level and alternate level are three level of scheduling. Priority will assigned

by job resource ratio then each queue is compared with priority. The higher throughput and less completion time can be achieved through this algorithm. El-Sayed T. El-kenawy et al. [51] proposed a RASA based scheduling algorithm. Select the jobs based on execution time instead of completion time in this algorithm. Shalmali Ambike et al. [52] proposed a non-preemptive priority queuing model based scheduling algorithm in which Cloud user performs the activities and the QoS requirements are achieved. S. Selvarani et al. [53] proposed a cost-based scheduling algorithm, that divides all Cloud consumer jobs depending on importance of every job into three different lists. The resource cost and computation performance can be measured and computation/communication ratio is improved. Ioannis A. Moschakis et al. [54] proposed a job migration and starvation handling based gang scheduling algorithm in which parallel jobs are scheduled.

2.2.3 Resource Allocation Policies

Cloud Computing evolves from Grid Computing, which is also regarded as its backbone and basic structure. Dispersion, heterogeneity and uncertainty of resources bring challenges to resource allocation, which cannot be satisfied with traditional resource allocation policies in Cloud circumstances. Many Researchers have put forward solutions for efficient resources allocation based on market mechanisms. CDA (Continuous Double Auction) is one of the common market mechanisms being used currently. It ensures high efficiency and effective coordination of resource allocation. Kennedy's paper [55] proves that web resource allocation based on CDA framework is effective. Yun Li's paper [56] presents a Cloud resource assignment policy based on CDA framework and Nash equilibrium to fulfil effective resource allocation in Cloud environment. It is important for both Cloud Computing infrastructure operators and service operators to guarantee the reliability of dynamic provision resource. QoS (Quality of Service) and effectiveness of dynamic resource that service providers should ensure depended on the reliability of dynamic resource providing. For this question, researchers have proposed strategies such as dynamic resource provision on the basic of the law of failure [61] and dynamic resource allocation based on credibility but they never considered key QoS requirements [57]. The efficiency of resources allocation will directly influence the performance of the whole Cloud environment. Since Cloud Computing has its own features, original resource allocation policy and scheduling algorithms for Grid Computing are unable

to work under this condition. To deal with this issue, Hua's paper [59] proposed an Ant Colony Optimization algorithm for resource allocation, in which all the characteristics in Cloud are considered.

2.2.3.1 Dynamic Resource Allocation Strategy Based on the Law of Failure

Hua's paper [59], Schroede's paper [60] and Sahoo's paper [61] present that through the study of log on the system failure, researchers found that the failure presents a strong time and spatial locality.

2.2.3.2 Dynamic Resource Assignment on the Basis of Credibility

The requests for resource under Cloud environment typically exhibit strong volatility. To ensure the credibility of dynamic resources without affecting its service efficiency, researchers propose a more credible dynamic resource provision strategy [57]. In the Cloud, resource allocation model based on CDA mechanism mainly includes Cloud resource providing agent, Cloud resource requirement agent, and information serving agent. They consult each other to achieve a balance on the price and the amount of resource for transaction. When entering or leaving a Cloud resource system, both the owner of the resource and the Cloud user need to be registered to the information serving agent [58]. The owner will set a price and allocate the resources through resource providing agent, while the user will allocate appropriate amount of resource through resource requirement agent to the jobs needed to be done.

2.2.3.3 Ant Colony Optimization Algorithm for Resource Allocation

Ant Colony Optimization (ACO) is an updated bionic optimization algorithm which is in simulation of ant foraging behaviour [59]. It is originated by M. Dorigo et al. [62] who were inspired by the research result of the group behaviour of real ants. ACO algorithm shows characteristics of rapidity, distribution and global optimization when solving complex optimization problems. And the rapidity of finding the optimal solution is due to the regenerative feedback mechanism of pheromone. ACO algorithm can find out computing resources in unknown network topology and select the most appropriate one or more resources to user's job until it meets user's requirements.

2.2.4 Job Scheduling Algorithm under Cloud Circumstance

Job scheduling of Cloud Computing refers to the process of adjusting resources between different resource users according to certain rules of resource use under a given Cloud environment [63]. Resource management and job scheduling are the key technologies of Cloud Computing. At present, there is no uniform standard for job scheduling in Cloud. Most algorithms focus on job dispatcher, which is almost responsible for all the task allocations, responses and retransmissions [64]. Over-reliance on the scheduler may lead to some Virtual Machines overload while others are idle after dispatcher allocating tasks according to the load of Virtual Machines. When this occurs, the only solution is to assign tasks for next period according to what the feedback scheduling device gets. The process in different Virtual Machines is independent. A Virtual Machine does not have access to other Virtual Machines' running conditions. One of the problems is the fluctuating workloads and the other one is the environment of Virtual Machines changes, resulting in the problems of some Virtual Machines and making them unable to send the information back to the scheduler [63]. This will also cause some Virtual Machines overload while others free.

2.2.4.1 Dynamic Scheduling Algorithm Based on Threshold

To get the real-time feedback of the state of Virtual Machine, there are two ways. One of them is to construct a set of feedback mechanism between dispatcher and Virtual Machines to get the real-time feedback of the tasks load on Virtual Machine, and then make a real-time adjustment on job allocation upon the fact of Virtual Machines. The other one is to use the dynamic scheduling among Virtual Machines themselves to get the real-time state of the load of Virtual Machines. If overload or idleness occurred, tasks could be readjusted and redistributed among Virtual Machines. By dynamic dispatch in Virtual Machines, dynamic scheduling algorithm based on threshold can allocate jobs and resources flexibly and reduce the efficiency impact caused by the synchronization among Virtual Machines [63]. However, task allocation between Virtual Machines refers to synchronization problems, which is also the biggest problem of the dynamic scheduling algorithm based on threshold. Since each Virtual Machine is independent to each other, in the other word they are non-interfering. They can perform tasks in parallel. If Virtual Machines are synchronized, they

inevitably bring effects to their performance. Therefore, the synchronization operation should be kept to minimum range.

2.2.4.2 Optimized Genetic Algorithm

Genetic Algorithm (GA) is proposed by Holand, who was inspired by biological evolution, in 1975. Parallelism and global solution space search are the two notable features of the GA [65]. On the basis of Map/Reduce model in Cloud [66], in order to cut down both the total running time and the average time of task execution, researchers have added one more fitness function to improve the GA.

2.2.5 Resource Distribution Policies

The input parameters to Resource Distribution Policies (RDPs) and the way of resource allocation vary based on the services, the type of applications which demand resources. The schematic diagram in Figure 2.1 depicts the classification of Resource Distribution Policies (RDPs) proposed in Cloud paradigm. The following section discusses the RDPs employed in Cloud [67] [68] [69] [70] [71] [72] [73] [74] [75] [76] [77].

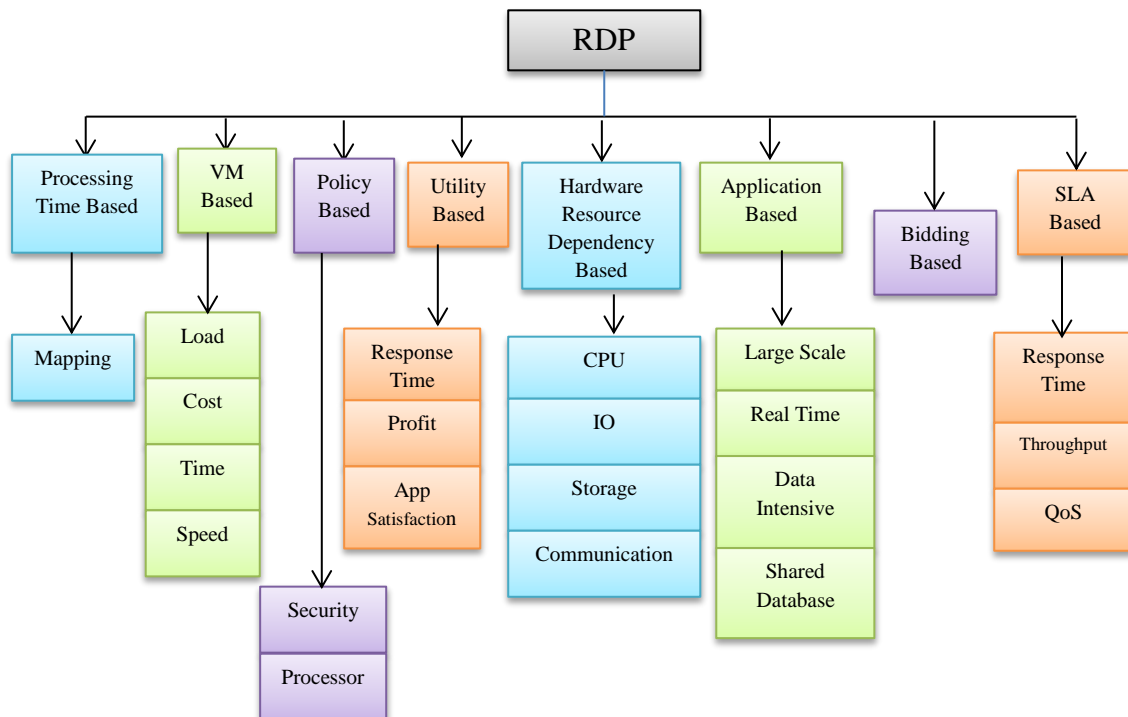


Figure 2.1: Resource Distribution Policies

- Processing Time Based

In the work by Jiani et al. [67], actual workload processing time and pre-emptible scheduling is considered for resource provision. It increases resource utilization by using different modes of leasing computing capacities and overcomes the problem of resource conflict. However estimation of processing time for every Cloud workload is a difficult for Cloud consumer.

- Policy Based

Since centralized user and resource management lacks in scalable management of users, resources and organization-level security policy, Dongwan et al. [68] has proposed a decentralized user and virtualized resource management for IaaS by adding a new layer called domain in between the user and the virtualized resources. Based on Role Based Access Control (RBAC), virtualized resources are allocated to users through domain layer. The most-fit policy allocates a job to the cluster, which creates an unused processor distribution, instantaneous successive job allocations is done. The clusters are assumed to be geographically distributed. The time overheads are insignificant compared to the system long time process but time complexity is increased.

- VM Based

The system composed of a virtual network of Virtual Machines capable of live migration across multi - domain physical infrastructure. A virtual computation environment is capable to automatically move itself through the infrastructure and scale its resources through dynamic availability of resources and dynamic application requirement. But the above work considers only the non-preemptable scheduling policy. This approach lacks in scalability due to the tasks are scheduled on fixed number of resources. Recent studies on allocating Cloud VMs for real time tasks focus on different aspects like infrastructures to enable real-time tasks on VMs and selection of VMs for power management in the data center. Karthik et al. [69] presented that the based on the speed and cost of different VMs in IaaS the resources are assigned. It permitting the Cloud consumer to select VMs and reduces cost.

- Utility Based

There are several suggestions that manage VMs dynamically by improving specific objective function such as reducing cost function, meeting QoS objectives

and decreasing cost performance function. The utility property is used to define the optimized objective function is selected based on revenue, response time, deadlines and number of QoSs achieved. There are few works [71] that dynamically allocate CPU resources to meet QoS objectives by first allocating requests to high priority applications. Dorian et al. [72] proposed utility based resource allocation for VMs through VM migration. By changing VM utilities the cost-performance trade-off is controlled. Considering CPU, memory and communication, the resource allocation is done based on response time as a measure of utility function for multi-tier Cloud Computing system.

- **Hardware Resource Dependency Based**

Multiple Job Optimization (MJO) scheduler is suggested for hardware utilization improvement. Jobs might be categorized based on hardware-resource dependency (Network I/O-bound, CPU-bound, Disk I/O bound and memory bound). The category of jobs and parallel jobs of different types are detected and resources are allocated based on the type. MJO primarily focus on I/O and CPU resource. The classic open source frameworks for virtualization management are Nimbus, Open Nebula and Eucalyptus [73]. The key characteristic of these frameworks is to assign virtual resources from a set of physical resources and with this a virtualization resource pool decoupled with physical infrastructure is formed. All these frameworks cannot support all the application modes due to complexity of virtualization technology.

- **Bidding Based**

Cloud resource allocation by auction mechanism based on closed-bid auction is addressed by Wei-Yu Lin et al. in [74]. The price is determined by collecting bid from all the Cloud consumers. The distribution of resources to the first i_{th} maximum bidders below the price of the $(i+1)_{th}$ maximum bid. Through this mechanism the Cloud service provider decision rule and allocation rule is made simplified by reducing the resource problem into ordering problem. Due to truth values the maximum profit cannot be achieved.

- **Application Based**

Resource allocation policies are addressed based on the type of the applications. Tram et al. [75] described that virtual infrastructure distribution policies are considered for workflow based applications where based on the workflow

representation of the application the resources are assigned. The application logic can be understood to create an execution schedule estimate for work flow based applications. To estimate the exact amount of resources that will be required for every execution of the application. To assign resources and schedule computing tasks, the Naive, FIFO, Optimized and services group optimization strategies are designed.

- SLA Based

In Cloud, the works related to the SaaS providers considering SLA are still in their infancy. Moreover Lee et al. [76] have addressed the problem of profit driven service request scheduling in Cloud Computing by considering the objectives of both parties such as service providers and consumers. But the author Linlin Wu et al. [77] have contributed to RAS by focusing on SLA driven user based QoS parameters to maximize the profit for Cloud providers. The mappings of Cloud consumer workload with available resources and policies used to minimize the cost by optimizing the resource allocation within a VM.

As per the above discussion, most of the work done in resource allocation area of Cloud is done in static allocation and mostly for homogenous compute nodes. The comparison of existing scheduling algorithms is summarized in Table 2.2.

Table 2.2: Comparison of Existing Scheduling Algorithms

Approach	Scheduling Criteria	Scheduling Aspects	Problems Discussed	Classification	Tool
Improved Cost-Based Algorithm for Task Scheduling [53]	Cost, Performance	Unscheduled task groups	1. Measures both resource cost and computation performance. 2. Improves the Computation/communication ratio.	Cloud	CloudSim
Scheduling Transaction-Intensive Cost-Constrained Cloud Workflows [78]	Execution Cost and Time	Workflow with large number of instances	1. To minimize the cost under certain user-designated Deadlines. 2. Enables the compromises of execution cost and time.	Cloud	SwinDeW-C
A Compromised Time-Cost Scheduling Algorithm [79]	Cost and Time	An array of workflow instances	1. It is used to reduce cost and cost.	Cloud	SwinDeW-C
A Particle Swarm Optimization-based Heuristic for Scheduling [41]	Resource Utilization, Time	Group of tasks	1. It is used for three times cost savings as compared to existing. 2. It is used for good distribution of workload onto resources.	Cloud	Amazon EC2

SHEFT Workflow Scheduling Algorithm [80]	Execution Time, Scalability	Group of tasks	1. It is used for optimizing workflow execution time. 2. It also enables resources to scale elastically during workflow execution.	Cloud	CloudSim
Market-oriented Hierarchical Scheduling Strategy [44]	Makespan, Cost, CPU Time	Service level scheduling ,task level scheduling	1. The overall running cost of Cloud workflow systems will be minimized. 2. It can be used to optimize both make span and cost simultaneously.	Cloud	SwinDeW-C
Multiple QoS Constrained Scheduling Strategy of Multi-Workflows [42]	Scheduling Success Rate, Cost and Makespan	Multiple Workflows	1. It is used to schedule the workflow dynamically. 2. It is used to minimize the execution time and cost.	Cloud	CloudSim
Optimal Workflow Based Scheduling (OWS) Algorithm [81]	CPU Utilization and Execution Time	Multiple Workflows	1. It is used to find a solution that meets all user preferred QoS constraints. 2. It is used to improve CPU utilization.	Cloud	CloudSim
RASA Workflow Scheduling [46]	Makespan	Grouped tasks	1. It is used to reduce make span.	Grid	GridSim

Authors that proposed dynamic resource allocation either they have considered private Cloud or those who have considered hybrid Clouds; have never provided the priority based resource allocation. The mechanism that provides adaptive resource allocation for pre-emptible jobs re-evaluate the tasks that are in the jobs submitted to certain Cloud but it does not consider the tasks that are assigned to that Cloud. The literature review shows that dynamic resource allocation is growing need of Cloud providers for more number of users and with the less response time. Hence the on-demand resource allocation based SLA as per defined task priority helps to satisfy the efficient provisioning of Cloud resources to multiple Cloud users.

2.3 Cloud Design Pattern Based Approaches

To analyse Cloud workload patterns, existing pattern based approaches have been discussed here. The simplest way to describe a pattern is that it delivers a proven solution to a common problem individually documented in a consistent format and commonly as part of a larger collection. Design patterns are patterns rotated around the design of automatic systems in the IT world. Design patterns [82] [83] [84] are helpful because of the following reasons:

- a) Represent field-tested solutions to common design problems.

- b) Organize design intelligence into a standardized and easily referenced format are generally repeatable by most IT professionals involved with design.
- c) Can be used to ensure consistency in how systems are designed and built can become the basis for design standards.
- d) Are usually flexible and optional (and openly document the impacts of their application and even suggest alternative approaches).
- e) Can be used as educational aids by documenting specific aspects of system design (regardless of whether they are applied).
- f) Can sometimes be applied prior and subsequent to the implementation of a system.
- g) Can be supported via the application of other design patterns that are part of the same collection.
- h) Enrich the vocabulary of a given IT field because each pattern is given a meaningful name.

The industry-driven evolution of Cloud Computing tends to complicate the common underlying architectural concepts of Cloud offerings and their implications on hosted applications. Design patterns are used for documentation of architectural principles and to make good solutions to reoccurring (architectural) Cloud challenges reusable [82]. To capture Cloud Computing best practice from existing Cloud applications and provider-specific documentation, an elaborated pattern format is used to enabling abstraction of concepts and reusability of knowledge in various use cases. Elasticity empowers Cloud users to reserve and release Cloud resources dynamically and based on the currently experienced workload [83]. In industry, Cloud providers already make use of pattern based descriptions and provide vendor-specific pattern formats and graphical notations. These patterns can be used to model Cloud applications hosted on these providers' Clouds. If a certain number of resources have been provisioned, a human system manager shall be notified via e-mail. The administrator has to approve further resource provisioning via a management user interface to control the costs generated by the application [84].

Design patterns will be used to tackle the complexity of the Cloud environment, a well-established concept to describe pieces of knowledge to capture architectural styles [84]. The patterns are interrelated in a structure manner and ensure an abstraction of implementation detail and use case specifics to describe reusable

solutions to reoccurring problems. Any researchers already proposed several patterns capturing Cloud architecture principles and management styles common for different Cloud providers [85]. Using just one format enables the categorization of Cloud providers regarding the patterns they support and sets the context for architectural patterns to be implemented by application developers. The use of a common pattern format, further, eases perception [86] and, thus, increases Cloud technology adoption. This pattern template is used to capture the knowledge from various experts.

- a) Presentation of information in a pattern format enables application architects to quickly absorb a large body of knowledge and to understand the impact that Cloud Computing, associated technologies, and products have on application architectures. The patterns, thus, capture how architectural decisions impact the functional and non-functional requirements of Cloud based applications [87].
- b) Cloud patterns provide abstract knowledge about Cloud products to application developers. A pattern set of distinct semantic of design pattern interrelations furthermore aids developers during the discovery and use of design patterns appropriate in their concrete use cases. Rapidly changing and evolving Cloud application implementations and available Cloud offerings may be easier coped with, because abstract template solutions, contained in the pattern descriptions, may be applied to different Cloud technologies [88].
- c) Abstract pattern-based descriptions of concepts can be used in education and ensure a longer persistence and relevance of obtained knowledge, because of their focus on abstracted information rather than on concrete implementations and offerings [89].

Pattern-based descriptions have been used frequently to describe good solutions to reoccurring problems during the architectural design of applications. G. Hohpe et al. [87] described patterns on how IT reconsolidation may reduce the environmental impact of the IT architecture by restructuring application architectures and supported processes. However, to be applicable in the context of Cloud Computing each pattern would have to be evaluated and possibly needs to be revised. In the field of Cloud Computing, patterns are gaining momentum. In industry, they are often used in a less formal way to present architectural guidelines [90]. Binz et al. [91] researched patterns for application migration.

2.3.1 Pattern Identification Process

From the literature [95] [96] [97], the identification process of design patterns is shown in Figure 2.2.

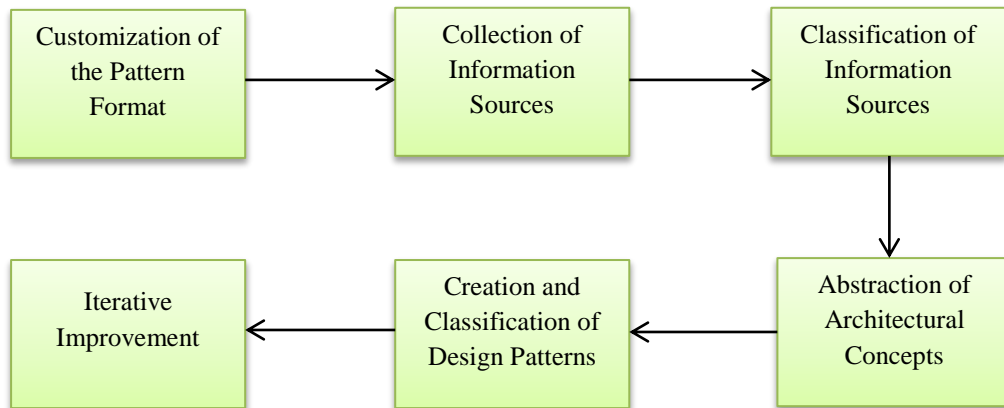


Figure 2.2: Pattern Identification Process [95]

- **Customization of the Pattern Format**

As shown in Figure 2.2, this format allows the omission of optional sections, which are then summarized with mandatory sections as indicated in the word template. Each pattern is identified by a unique name and has an icon associated with it. Then, a driving question states the problem solved by the pattern to enable readers to quickly check if the pattern fits to problems at hand. The expression of Cloud types and Cloud offerings in the pattern format has proven to be very valuable [84].
- **Collection of Information Sources**

In this step, different information sources are collected. An excel spread sheet contained in the pattern will be used. For each information source, general information, such as document names, hyperlinks etc. are collected. Further, the Cloud Computing relevant information obtained from the sources is summarized. This summarization should be expressed respecting the context of the information source and its domain specific terms. Examples for information sources are provider guidelines, knowledge about existing applications, books, and journals [91].
- **Classification of Information Sources**

The categorization of architecture domain is done from brief knowledge mined from the different information sources. During the classification process the

summaries created for information sources may be split up into multiple entries to allow a fine granular classification [86].

- **Abstraction of Architectural Concepts**

Till now, the brief knowledge mined from the several information sources remains related to Cloud provider. It, consequently, desires to be distracted to essential architectural concepts. During this step, summaries may again be split up. This allows a grouping of summarized knowledge regarding the abstracted concepts [85] [92].

- **Creation and Classification of Design Patterns**

Based on the abstract concepts, design patterns are now compiled from the information sources. Meanwhile the design patterns are not designed, then known and exposed from present solutions, confirming an appropriate level of abstraction is always a challenge [89].

- **Iterative Improvement**

Research is shown that patterns evolve continuously through discussion in a community. This community should be heterogeneous, thus, should contain experts and beginners alike to ensure that obtained patterns contain all necessary information while remaining understandable. As per design patterns are generated by several different contributors, an arrangement of graphical components can be pursued during the revision for a more a consistent look-and feel. Therefore, during iterative improvement, graphical designers are involved to obtain easy to understand graphics [91].

2.3.2 Problems in Patterns

The classification of problems [90] [91] that will be discussed through design pattern into the following domains is based on this information:

- a) **Accounting & Controlling** - The use of pay-per-use billing is inherent to Cloud Computing. Additionally, providers may offer long term reservation of resources at a lower price, such as Amazon Reserved Instances [88]. These are intended for the static workload of customers. Variable pricing depending on the overall utilization of a Cloud are also available, such as Amazon Spot Instances. These billing models should be reflected in the application architecture to reduce runtime costs.

- b) Application Migration - Many existing applications could benefit from Cloud Computing. To move applications to the Cloud, the applications' infrastructure requirements and how they are respected by the application architecture need to be reconsidered [84].
- c) Cloud Integration - When using Cloud Computing, a company often faces the challenge that different computing environments have to be integrated. For example, legacy applications may reside in on-premise data centers, whereas others use a public Cloud [89].
- d) Data Storage - The inconsistency of data replicas is well-researched in distributed systems [87], Cloud Computing has shown how this concept may be employed to increase the availability and performance of an application [92] to handle large amounts of data and users efficiently.
- e) License Management - Licenses issued per CPU [91] executing software are hard to measure or enforce in the Cloud where hardware virtualization is inherent. This renders the use of many software products employing this licensing complicated to impossible in the Cloud. Since virtual servers may be started and stopped dynamically in the Cloud, best practices for license management are required to ensure a compliant deployment of Cloud resources, for example, during automated scaling processes.
- f) Monitoring, Analysis, and Reporting - Cloud Computing is often considered for the aspect of profitable. The presented resource sharing, yet, can also be exploited for energy savings. In either case, information about Cloud resources has to be collected and analysed [89]. Sharing resources between different projects and products significantly complicates such computations.
- g) Multi-Tenant Cloud Middleware Sharing - Cloud resources between multiple Cloud users, also called tenants, are a very important concept to benefit from economies of scale [85]. Patterns are, therefore, required in this domain to address challenges, such as tenant-isolation, version management, customer-specific application customization, migration of tenants between software versions and runtime environments etc.
- h) Security and Compliance - Many of the security issues found in non-Cloud applications are also arising in a Cloud environment. These concepts have been captured in a pattern format [86] and may be applicable in Cloud Computing as well. New security threats arise from malicious use of dynamic

Cloud resources or from other Cloud users. Regarding compliance, Cloud Computing introduces the challenge that a company's IT has to compete with public Cloud services. Methods how to address these issues have been investigated but there are currently no practical solutions to extract patterns from.

2.3.3 Pattern Terminology

From the literature [87], the pattern terminology of design patterns are given below:

- a) Patterns for Feedback - The feedback of readers new to the Cloud domain was especially useful during step 6 of the pattern identification process (iterative improvement), because it is difficult for experts to find an adequate level of abstractions during the initial identification and description of patterns.
- b) Transparency of Pattern Identification - The presented pattern identification process enables transparency of this task and allows other to retrace the steps undertaken to identify a pattern.
- c) Categorization of Cloud Providers based on Design Patterns - Cloud design patterns enabled a classification of providers regarding the patterns supported by them.
- d) Patterns for Documentation - Generally, patterns can be used in software documentation to reduce its size and increase its quality [86]. Use of patterns in documentation increased readability, because the well-known concept of a pattern could be perceived easier and quicker while reducing the amount of introductory text required in documents.
- e) Pattern Vocabulary - The standard vocabulary is very useful to bridge different product user communities.

2.4 Data Mining Algorithms

Data mining techniques are being used for making decisions based on some specified rules. The three well known data mining classifier algorithms namely ID3, J48 and Naive Bayes are used for data mining [93] [94] [95]. Among them Naive Bayes is one of the most effective inductive learning algorithms used to make decisions based on predefined rules in decision trees is concerned [97].

2.4.1 Classification Algorithms

Systems that create classifiers are one of the frequently used tools in data mining. Such systems take a group of different cases as input; every one belongs to a small amount of classes defined by a stable set of attributes and output a classifier that can truthfully forecast the class to which a new case belongs. Datasets can have nominal, numeric or mixed attributes and classes. Not all classification algorithms perform well for different kinds of attributes, classes and for datasets of different dimensions. In order to design a standard classification tool, one should consider the behaviour of various existing classification algorithms on different datasets. The choice of a classification algorithm depends on the desires and the nature of classification required [97].

2.4.2 Decision Trees

Decision tree introduction has been considered in details in both areas of pattern recognition and machine learning [96]. This creates the experience expanded by individuals working in the region of machine learning and describes a computer program called ID3, which has evolved to a new system, named C4.5 (An enhanced version of C4.5 is J48) [94] [97].

2.4.2.1 ID3 Algorithm

ID3 is one of the well-known Inductive Logic Programming procedures, developed by Quinlan [93]. It is basically an attribute based machine-learning algorithm that builds a decision tree based on a training set of data and an entropy measure to build the leaves of the tree [94]. The informal formulation of ID3 is as follows:

- Define the element that has the highest statistics gain on the training set.
- Use this element as the root of the tree; generate a branch for each of the values that the attribute can take.
- For every branch, repeat this process with the subset of the training set that is categorized by this branch.

2.4.2.2 J48 Algorithm

J48 (enhanced version of C4.5) is based on the ID3 algorithm developed by Ross Quinlan [95], with additional features to address problems that ID3 was unable to

deal. In exercise, C4.5 uses one successful process for finding high correctness guesses, based on snipping the instructions dispensed from the tree constructed during the learning phase. Conversely, the principal disadvantage of C4.5 rule sets is the amount of CPU time and memory they need [95]. Given a set S of cases, J48 first grows an initial tree using the divide-and-conquer algorithm as follows:

- If all the different cases in C belong to the similar class or C is small, the tree is leaf labelled with the most frequent class in C .
- Otherwise, select a test based on a single attribute with two or more results. Create this test as the root of the tree with one branch for every result of the test, partition C into corresponding subclasses C_1, C_2, \dots according to the result for every different case, and apply the same technique recursively to every subclass.

There are usually many tests that could be chosen in this last step. J48 uses two heuristic criteria to rank possible tests: statistics gain, which reduces the total entropy of the subclasses $\{C_i\}$ and the default gain ratio that distributes statistics gain by the information provided by the test outcomes. After the building process, each attribute test along the path from the root to the leaf becomes a rule antecedent (precondition) and the classification at the leaf node becomes the rule consequence (post condition) [94]. To illustrate the post pruning of the rules, let us consider the following rule generated from the tree:

IF Condition THEN Conclusion

This rule is pruned by removing any antecedent whose removal does not worsen its estimated accuracy.

2.4.2.3 Naive Bayes Algorithm

Naive Bayes [98] is an extension of Bayes theorem in that it assumes independence of attributes. This supposition is not stringently accurate when considering grouping based on text extraction from a document as there are relationships between the words that collect into concepts. Problems of this kind, called problems of supervised classification, are ubiquitous. It is simple to construct without any requirement for complex iterative parameter approximation patterns [96]. This means it may be enthusiastically applied to huge data sets. It is robust, easy to interpret, and often does surprisingly well though it may not be the best classifier in any particular application.

The Figure 2.3 shows the experimental results using decision trees with the ID3 and J48 (extension of C4.5) algorithms [97], along with the results obtained from Naive Bayes algorithm.

Precision Rate - It is the fraction of retrieved Information that is relevant to the search.

$$\text{Precision Rate} = \frac{\text{Relevant Information} \cap \text{Retrieved Information}}{\text{Retrieved Information}}$$

Recall Rate - Recall in information retrieval is the fraction of the Information that are relevant to the query that are successfully retrieved.

$$\text{Recall Rate} = \frac{\text{Relevant Information} \cap \text{Retrieved Information}}{\text{Relevant Information}}$$

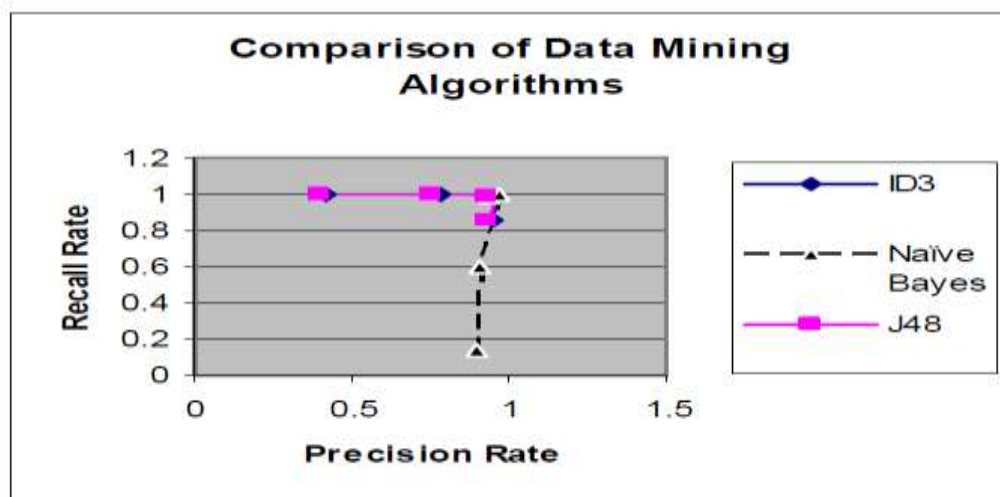


Figure 2.3: Precision-Recall Characteristics [97]

In Table 2.3, comparison is done with respect to time taken to build the model, along with the error rate for all the three data mining algorithms.

Error rate - The degree of errors encountered during extract information from a data set and transforms it into an understandable structure.

Table 2.3: Comparison of Data Mining Algorithms [97]

Experiments	Overall Error Rate	Time taken to build the Model in Seconds
Naive Bayes	3.46%	0.11
J48	3.47%	0.88
ID3	3.47%	1.8

Comparison of data mining algorithms, as shown in Figure 2.4 suggest that the ability of the various classification methods examined could be considered as good, i.e. the classifier stability is very strong. Root mean square error in the same way suggests that the error rate is very small, which can be considered as a measure of effectiveness of the model. It can be observed from the Figure 2.4, that the accuracy of overall classification for Naive Bayes [98] for all classes is better than the overall accuracy obtained in the case of J48 and ID3 algorithms. It is observed from the results obtained by experimentation that the Naive Bayes model is quite appealing [97] because of its simplicity, elegance, robustness and effectiveness.

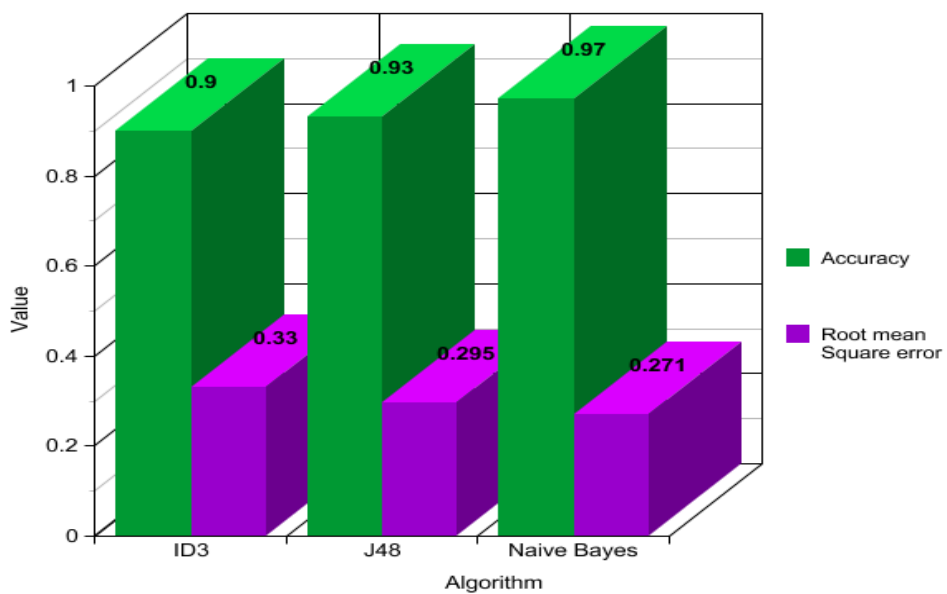


Figure 2.4: Comparison of Data Mining Algorithms [97]

2.5 Conclusion

Different factors related to Cloud workloads have been discussed in this chapter. The various existing resource allocation algorithms have been identified and compared. Based on the literature survey the different design patterns have been identified and discussed. Different data mining techniques have been compared based on different criteria and the best data mining technique that has been used to make the decision to choose scheduling criteria has been discussed. Next chapter focuses on the analysis of the problem based on the literature review.

Based on literature survey, the problem of allocation of resources to the Cloud workloads has been identified. This chapter presents the problem statement and objectives and commitments of this research work.

3.1 Problem Statement

To map the best resource to a corresponding Cloud workload is a tedious task and the problem of finding the best resource – Cloud workload pair according to Cloud consumer application requirements is an optimization problem [19]. The resources and Cloud workloads can leave and join the Cloud dynamically. Cloud resources are heterogeneous and dynamic in nature. In this work, independent Cloud workloads have been considered to handle the realistic scenarios as there are many scenarios in which the need of scheduling Cloud workloads arises. Firstly, this problem is suitable to Cloud systems because of the nature of Cloud customers, who submit Cloud workloads in an independent manner to the system. Secondly, Cloud systems are most useful for massive parallel processing, in which large amounts of data are processed independently. In this work, the scheduling of workloads has been considered from both the Cloud customer and Cloud provider's point of view. The user wants to minimize the cost whereas the Cloud provider wants to minimize the execution time and energy consumption. In this problem, the most popular and extensively studied optimization criteria, i.e. the minimization of the execution time has been considered. Execution time is used to indicate the general productivity of the Cloud systems. Smaller values of execution time and energy consumption indicate that the scheduler is planning the Cloud workloads in an efficient manner. Cost is another optimization criterion, which refers to the total cost of the Cloud workload execution on a particular resource. The problem has been derived to get an optimal solution. The problem can be expressed as: To consider this problem, a set of independent Cloud workloads $\{w_1, w_2, w_3, \dots, w_m\}$ to map on a set of heterogeneous and dynamic resources $\{r_1, r_2, r_3, \dots, r_n\}$ has been taken. $R = \{r_k | 1 \leq k \leq n\}$ is the collection of resources and n is the total number of resources. $w = \{w_i | 1 \leq i \leq m\}$ is the collection of Cloud workloads and m is the total number of Cloud workloads. The estimated

time to compute the value of each Cloud workload on each resource is assumed to be given by the consumer-supplied information, experimental data, Cloud workload profiling and analytical benchmarking.

3.2 Objectives and Commitments

The main objective of this work is to analyse the Cloud workloads, cluster them on the basis of common patterns and then provision the Cloud workloads before actual scheduling. This helps in better matchmaking and efficient scheduling. Further scheduling has been done based on scheduling policies and their corresponding algorithms. The following are main objectives of this work:

- a) Clearly understand the current and potential future requirements and expectations of the resource consumers.
- b) Develop Cloud workload management framework that aims at minimizing the energy consumption and at the same time meet the Cloud workload deadline.
- c) Based on workload details, identify the scheduling criteria (Compromised Cost - Time Based, Time Based, Cost Based and Bargaining Based) with the help of decision tree.
- d) Implement the scheduling algorithms in CloudSim 3.0 and validate scheduling criteria through the experimental results.

3.3 Conclusion

This chapter discussed the problem statement and objectives and commitments. Next chapter discusses the Proposed Framework for Cloud Workloads.

Proposed Framework for Cloud Workloads

This chapter presents the design of proposed solution through a Cloud Workload Management Framework. Framework comprises of Cloud Workload Management Portal (CWMP), Bulk of Workloads, Cloud Workload Analyser, Policy Selector, Scheduler, Resource Information and Resource Pool. Framework is based on four resource scheduling policies (Compromised Cost - Time Based (CCTB) Scheduling Policy, Time Based (TB) Scheduling Policy, Cost Based (CB) Scheduling Policy and Bargaining Based (BB) Scheduling Policy).

4.1 Cloud Workload Management Framework

Need of optimized resource scheduling in Infrastructure as a Service (IaaS) can be achieved using the proposed framework. The proposed Cloud Workload Management Framework is shown in Figure 4.1. The framework compromises of following units:

4.1.1 Cloud Workload Management Portal

The Cloud workload details are gathered through the Cloud Workload Management Portal (CWMP) from Cloud consumer. Web browser acts as an interface for both consumer and provider. The Cloud provider generates the workload schedule based on the workloads details specified by the user. The workload generated by Cloud provider is based on the four resource scheduling polices, to allocate the resources to the Cloud workloads efficiently.

4.1.2 Bulk of Workloads

The number of Cloud workloads submitted by the Cloud user is processed in the queue. Based on the details given by user, the resources are assigned to the Cloud workloads for their execution.

4.1.3 Cloud Workload Analyser

The aim of Workload Analyser is to look at different characteristics of a Cloud workload to determine the feasibility of porting the application in the Cloud. The different Cloud workloads have different set of requirements and characteristics. This analysis also provides input to execution method, Cloud service choice and a preliminary business worth valuation (Cost benefit analysis).

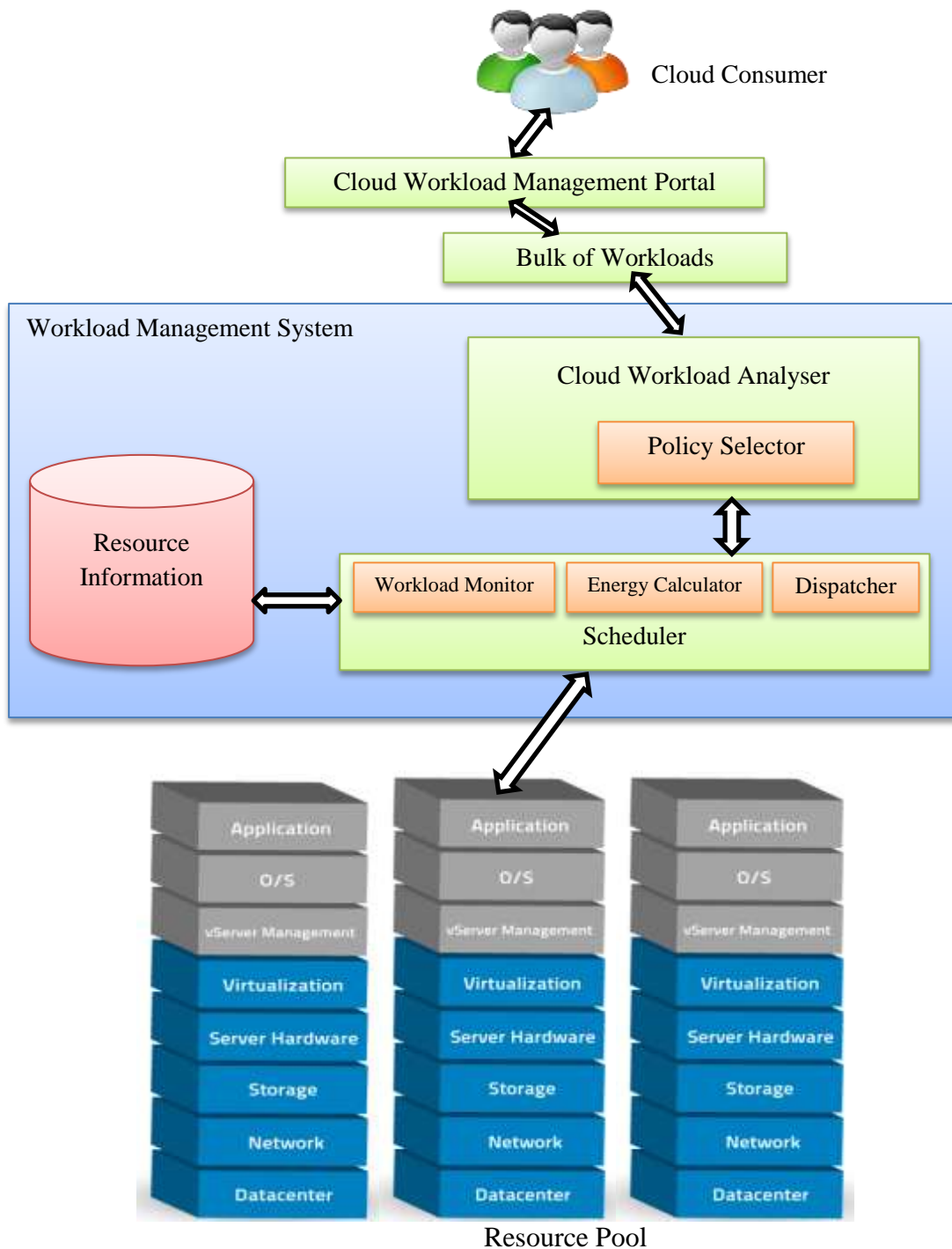


Figure 4.1: Cloud Workload Management Framework

4.1.4 Policy Selector

Four resource scheduling policies (Compromised Cost - Time Based (CCTB) Scheduling Policy, Time Based (TB) Scheduling Policy, Cost Based (CB) Scheduling Policy and Bargaining Based (BB) Scheduling Policy) are proposed in this thesis. Decision tree is used to select the appropriate policy based on workload details described by Cloud consumer.

4.1.5 Scheduler

Scheduler is used to schedule the Cloud workloads and map the Cloud workloads with available resources based on the policy defined by user. Scheduler uses minimum number of resources to serve the Cloud workloads within specified budget and deadline. Energy is also calculated and compared with threshold energy value at different value of resources. The ¹workload is dispatched only, if the workload is executed within described budget and deadline and actual energy consumption is less than the threshold energy value.

4.1.6 Resource Information

The resource details include the number of CPU using, size of memory, cost of resources, type of resources and number of resources. All the common resources are stored in resource pool and reserve pool contains some reserve resources.

4.1.7 Framework Description

A flowchart depicts the flow of Cloud workloads from CWMP to resource allocation, is as shown in Figure 4.2. Based on Cloud consumer details CWMP assigns resources and executes Cloud workloads. During execution of a particular Cloud workload, the CWMP will check the current workload. If the Required Resources (RR) is more than the Provided Resources (PR) then it will check the scheduling policy in the following ways:

- If the scheduling policy is Time Based (TB) then it will reschedule the allocated resources to the Cloud workloads, otherwise allocate the resources to new Cloud workloads from reserve resource pool.
- If the scheduling policy is Cost Based (TB) then Cloud Workload Management Portal will ask to change the policy to execute Cloud workloads and pay the amount as required.
- If the scheduling policy is Bargaining Based (BB) then the Cloud workloads will be executed by negotiation or mutual agreement between Cloud consumer and Cloud provider.
- If the scheduling policy is Compromised Cost-Time Based (CCTB), Cloud provider will minimize cost and execution time. For successful execution of a Cloud Workload, the Actual Energy Consumption (E_{Cloud}) should also be less than Threshold Energy Value ($E_{Threshold}$).

¹ The terms “workload” and “Cloud workload” are synonymous and interchangeable.

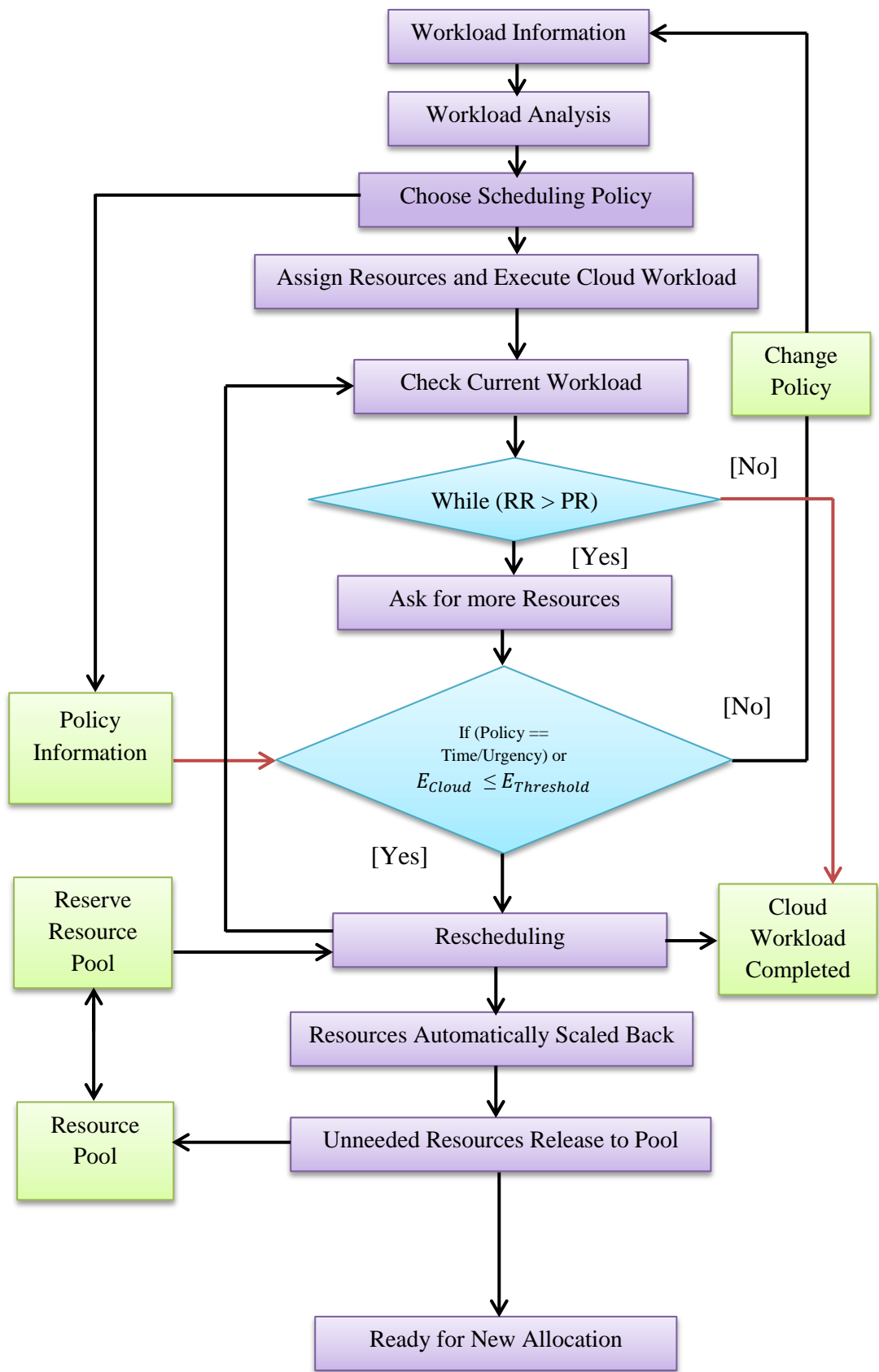


Figure 4.2: Flowchart of Proposed Framework

After successful execution of Cloud workloads, Cloud Workload Management Portal releases the free resources to resource pool and CWMP is ready for execution of new Cloud workloads.

4.2 Framework Assumptions

The proposed framework is resulting from the following assumptions of Cloud customer needs:

- Cloud Consumer wants “price” to be less and under budget. A regular classification of price here is the quantity of expenses; total payment or cost incurred to finish all of the workloads under consideration.
- Cloud Consumer wants “time” for completing all of the workloads to be less. If the dispatched workloads of a BoW (Bulk of Workloads) among a large number of resources, the latest completion time among all resources is the makespan of that workload and this completion time must be decreased as much as possible.
- Reducing the dynamic energy consumption by lowering the supply voltage at the cost of performance degradation. Develop resource management and scheduling algorithm that aim at minimizing the energy consumption and at the same time meet the Cloud workload deadline too.
- Cloud Customer prefers a “stable” workload assignment. Minimize unnecessary resource thrashing to minimize communication overheads.
- Cloud Customer prefers an “urgency” workload assignment. Urgency refers to the “minimise” execution time of a particular assignment.
- This framework assumes a single IaaS provider with uniform price list of the resources is being considered.
- If there is “urgency” then there is no need to move the workload in to workload queue, process directly by using reserve resource pool.

4.3 Framework Constraints

For Cloud Workload Management Framework, the following constraints have been considered:

1. Each Cloud workload to be scheduled for application's execution has a unique workload id.
2. Cloud workloads are independent.
3. Arrival of Cloud workloads for execution of application is random and Cloud workloads are placed in a queue of unscheduled Cloud workloads.
4. The computing/processing speed of the resources is measured in Multiple Instructions Per Second (MIPS) as per the Standard Performance Evaluation Corporation (SPEC) benchmark.
5. The processing requirement of a Cloud workload is measured in Million Instructions per Second (MIPS).
6. Execution time for every Cloud workload on a resource is obtained from objective function.

4.4 Framework Requirement Analysis

Requirement analysis of proposed framework is represented through UML diagrams.

4.4.1 UML Diagrams

Unified Modelling Language (UML) is a standard language for specifying, visualizing, constructing and documenting the artifacts of software systems, business modelling and other non-software systems. The UML uses mostly graphical notations to express the design of software projects. These graphical notations combined to construct UML diagram. Each UML diagram is designed to let developers and customers view a Cloud Workload Management Portal from a different perspective and in varying degrees of abstraction.

i. Use Case Diagram

Use Case diagram describes the behaviour of the CWMP from an external point of view. Use Case describes the core of the actual requirements. Figure 4.3 shows the Use Case diagram of the CWMP. It shows that member of the Cloud Workload Management Portal can ask the query to administrator, after doing the login member can check the energy consumption of both homogeneous and heterogeneous workloads and enter the Cloud Workload details. Administrator can accept or reject the request of a new user and answers the query asked by the Cloud consumer. After that Cloud provider generates the workload schedule based on consumer requirements and Cloud consumer can view workload schedule.

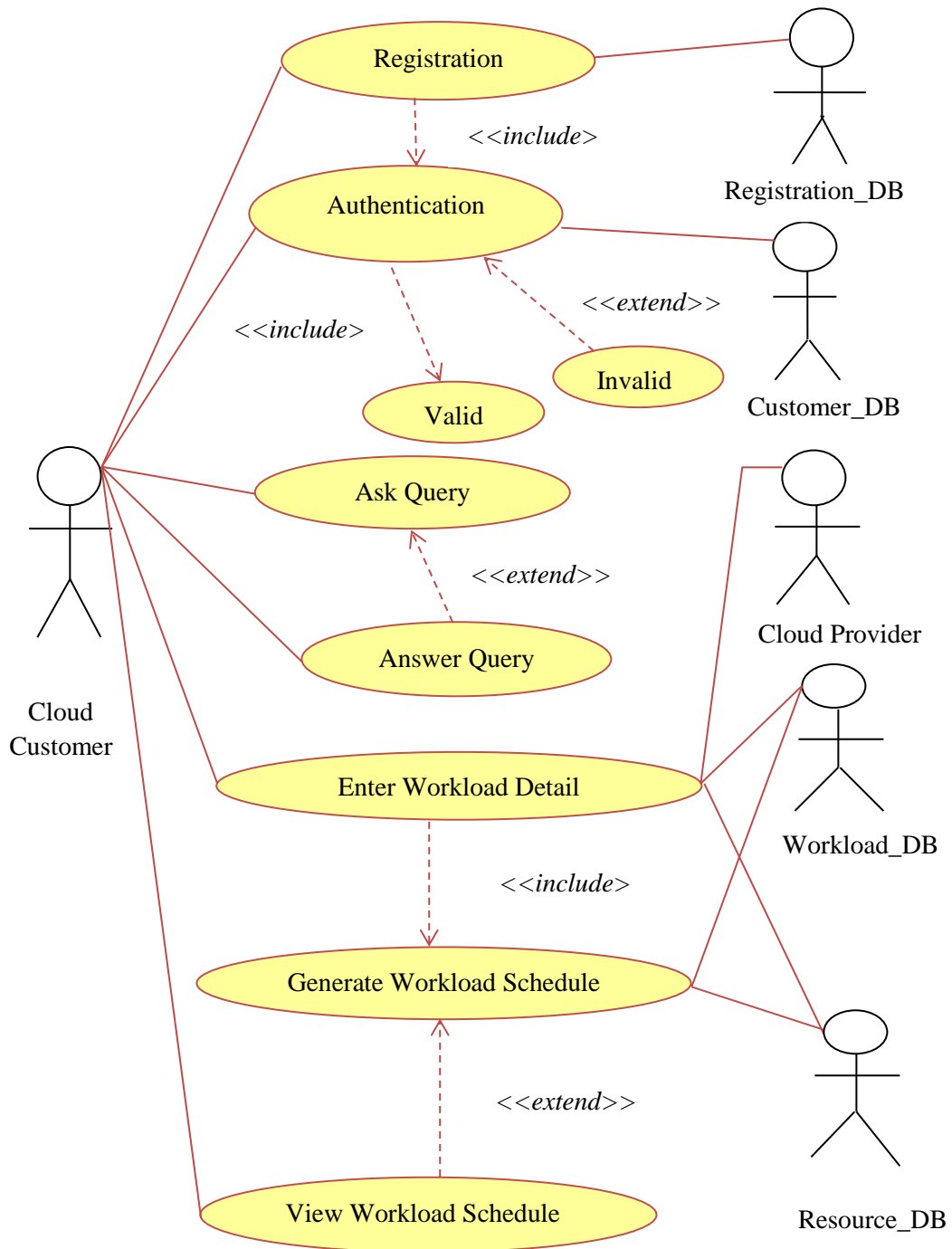


Figure 4.3: Use Case Diagram of the CWMP

ii. Sequence Diagram

Sequence diagram models the collaboration of objects based on a time sequence. It shows how the objects interact with others in a particular scenario of Use Case. Figure 4.4 shows the Sequence diagram of the CWMP.

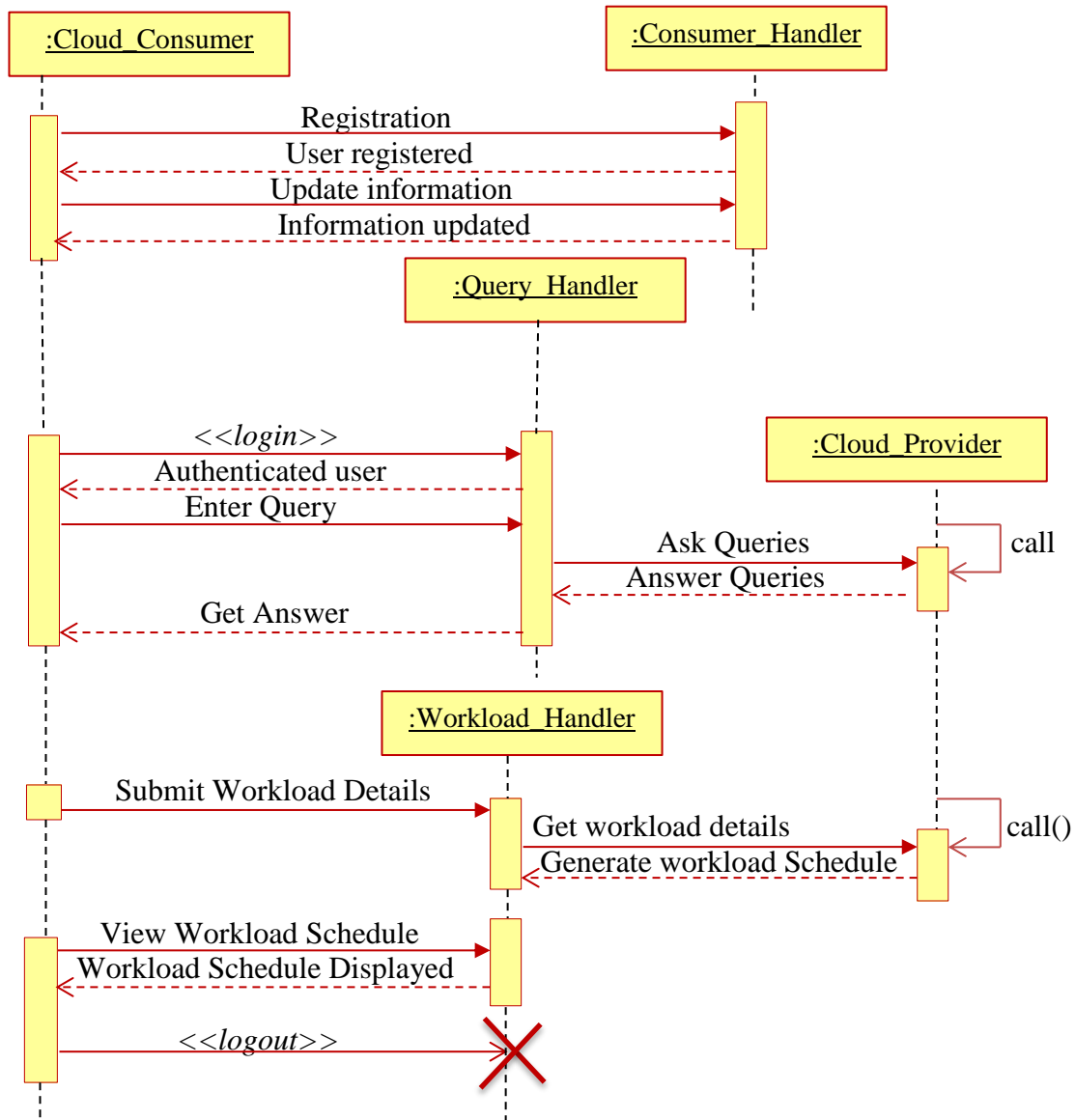


Figure 4.4: Sequence Diagram of the CWMP

iii. Activity Diagram

Activity diagram in UML is used for modelling the dynamic aspects of the Cloud Workload Management Portal (CWMP). An Activity diagram is essentially a flowchart, showing flow of control from activity to activity. For the most part, this involves modelling the sequential steps in a computational process. With an Activity diagram, model the flow of an object as it moves from state to state at different points in the flow of control. An Activity is an on-going non-atomic execution within a state machine. Activities ultimately result in some action, which is made up of executable atomic computations that result in state of the Cloud Workload Management Portal. Activity diagram of CWMP is shown in Figure 4.5.

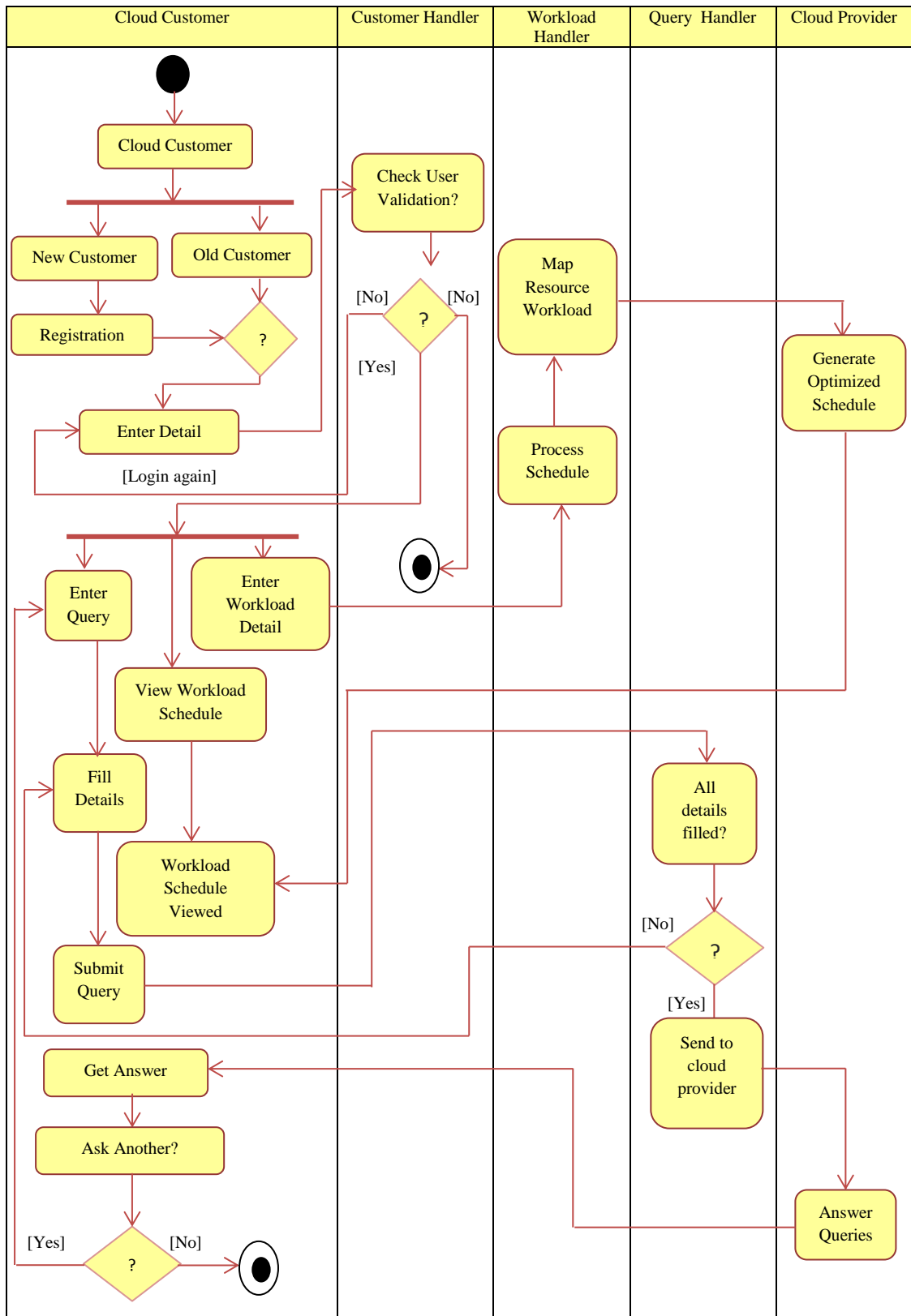


Figure 4.5: Activity Diagram of the CWMP

iv. Class Diagram

A Class diagram shows a set of classes, interfaces, and collaborations and their relationships. Class diagrams are the most common diagram found in modelling

object-oriented systems. Class diagram is used to illustrate the static design view of a CWMP. Class diagrams that include active classes are used to address the static process view of a CWMP. Class diagram is used to visualize the static aspects of CWMP is shown in Figure 4.6.

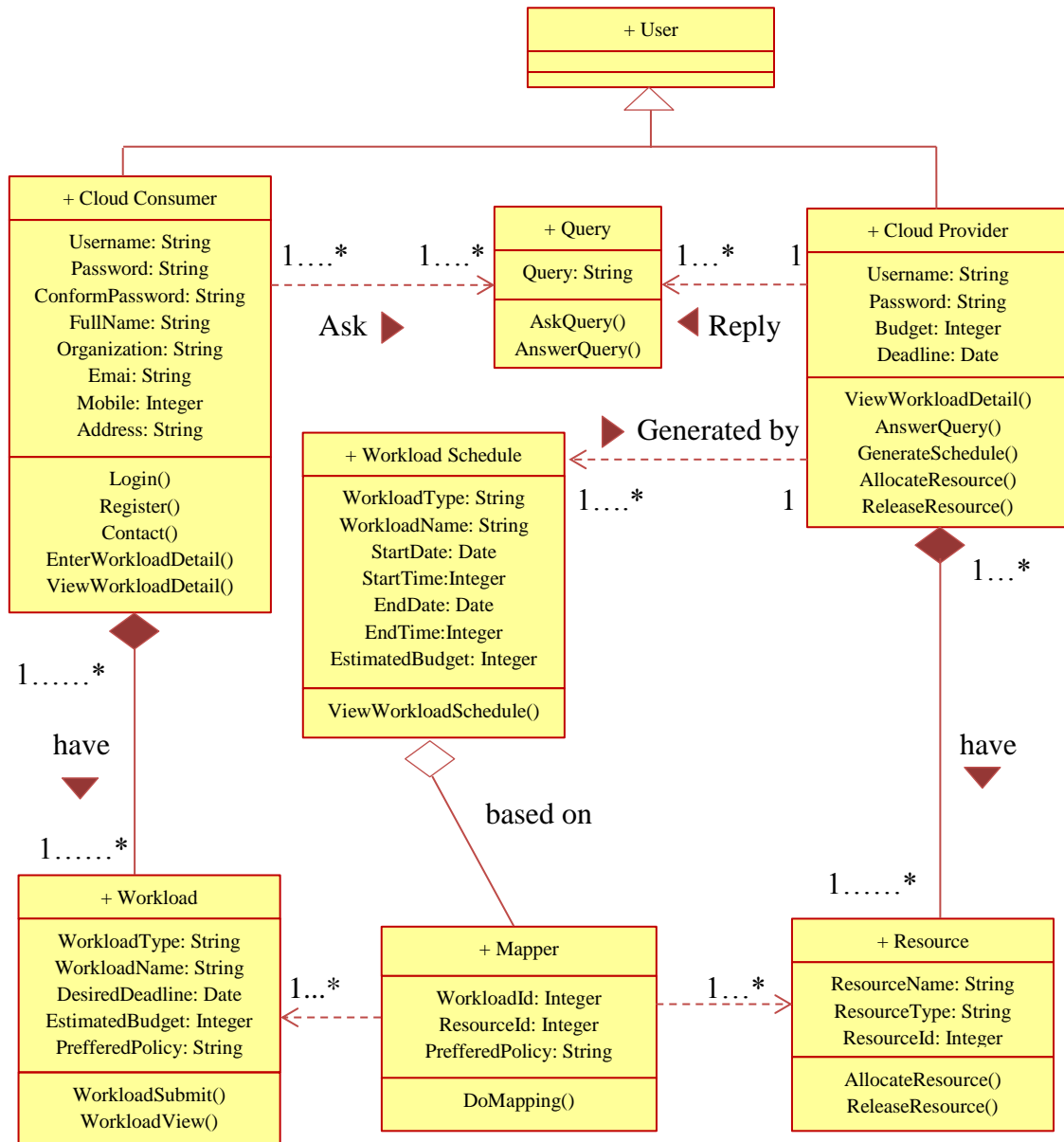


Figure 4.6: Class Diagram of the CWMP

v. State Chart Diagram

A State Chart diagram is emphasizing the flow of control from state to state. State Chart diagram is used to model the dynamic aspects of CWMP as shown in Figure 4.7. A state is a condition in the life of an object during which it satisfies some condition, performs some action. A transition is a relationship between two states indicating that an object in the first state will perform certain actions and enter the

second state when a specified event occurs and specified conditions are satisfied. Where CC denotes the Cloud Consumer.

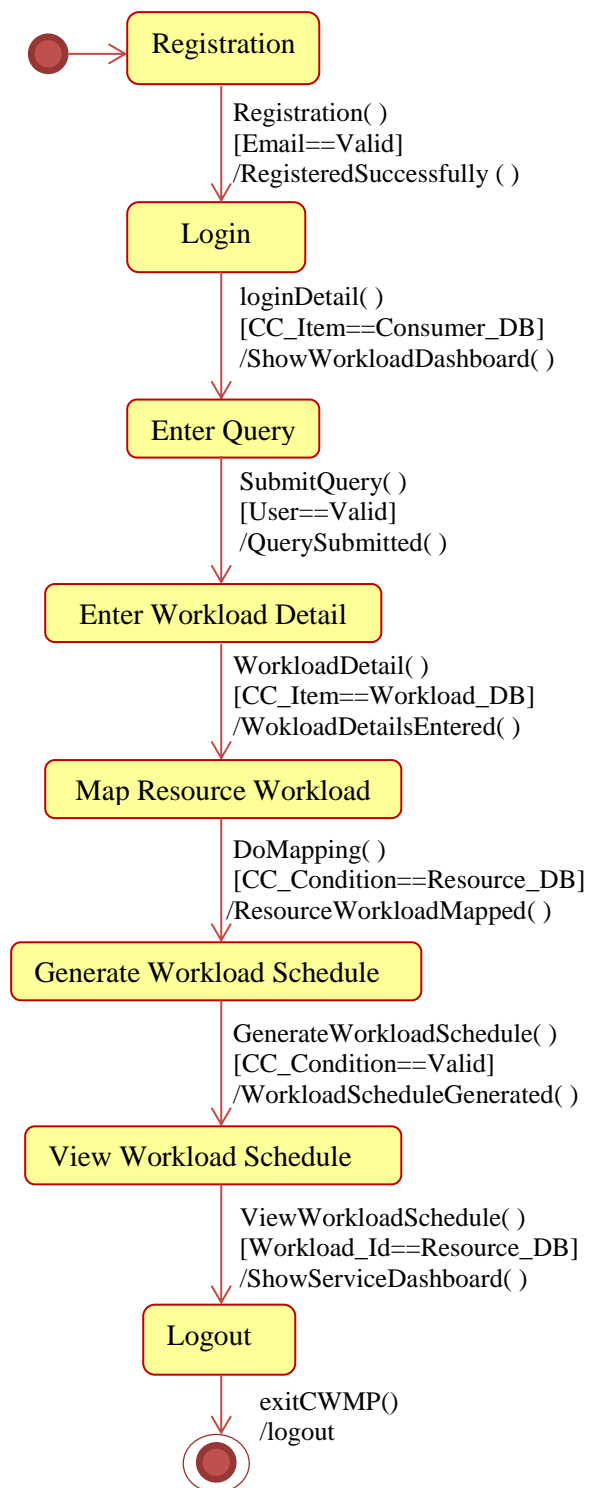


Figure 4.7: State Chart Diagram of the CWMP

vi. Collaboration Diagram

A Collaboration diagram emphasizes the structural organization of the objects that send and receive messages; a diagram that shows interactions organized around instances and their links to each other as shown in Figure 4.8.

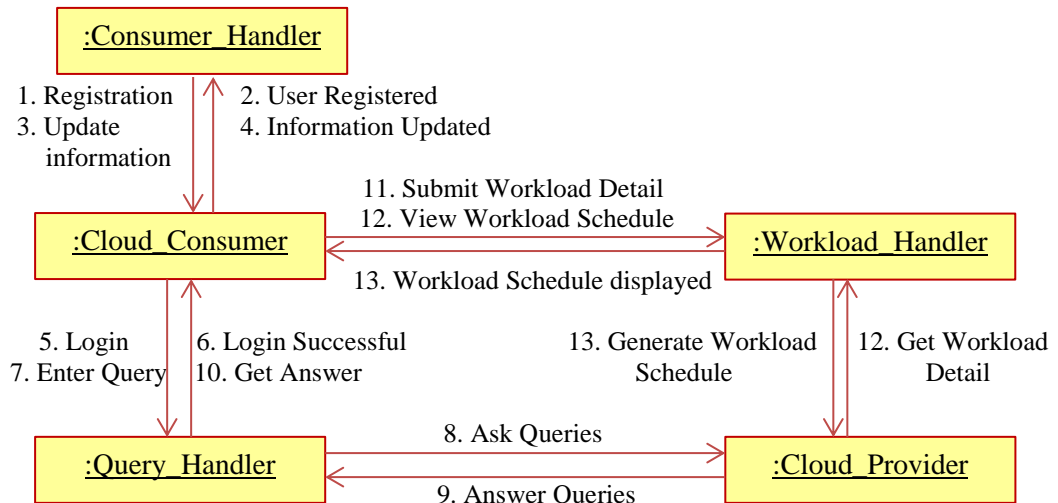


Figure 4.8: Collaboration Diagram of the CWMP

vii. Component Diagram

A Component diagram shows the organization and dependencies among a set of components as shown in Figure 4.9. A component is a physical and replaceable part of a system that conforms to and provides the realization of a set of interfaces.

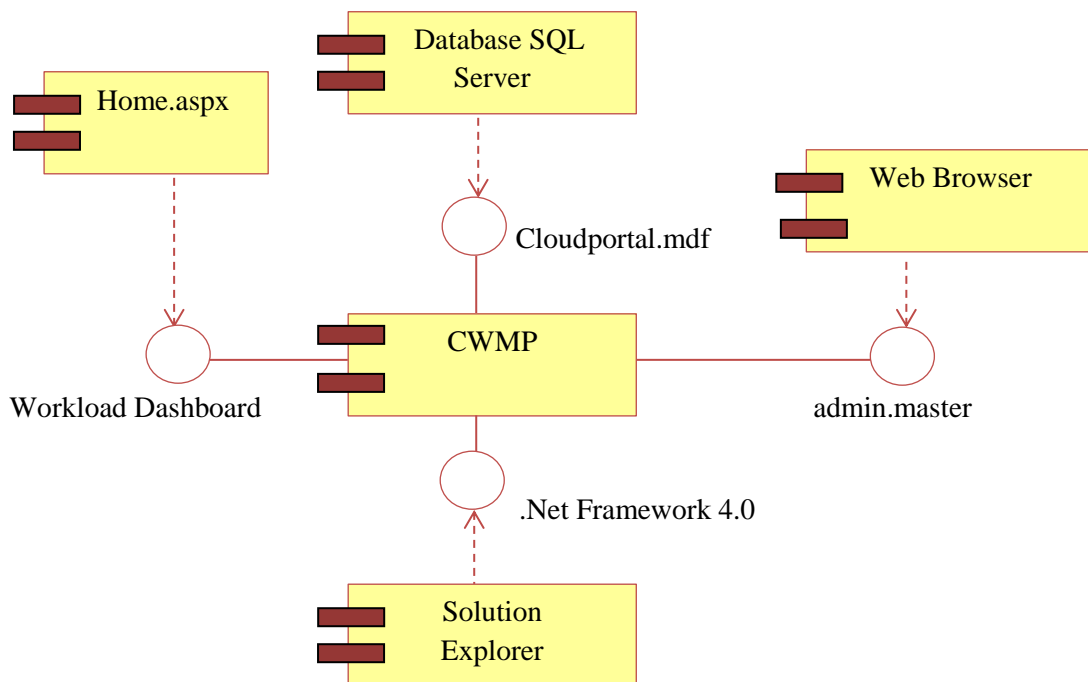


Figure 4.9: Component Diagram of the CWMP

To model a system's source code, model the compilation dependencies among these files using dependencies. Executable files, source code files, inbuilt libraries and databases are shown in Figure 4.10.

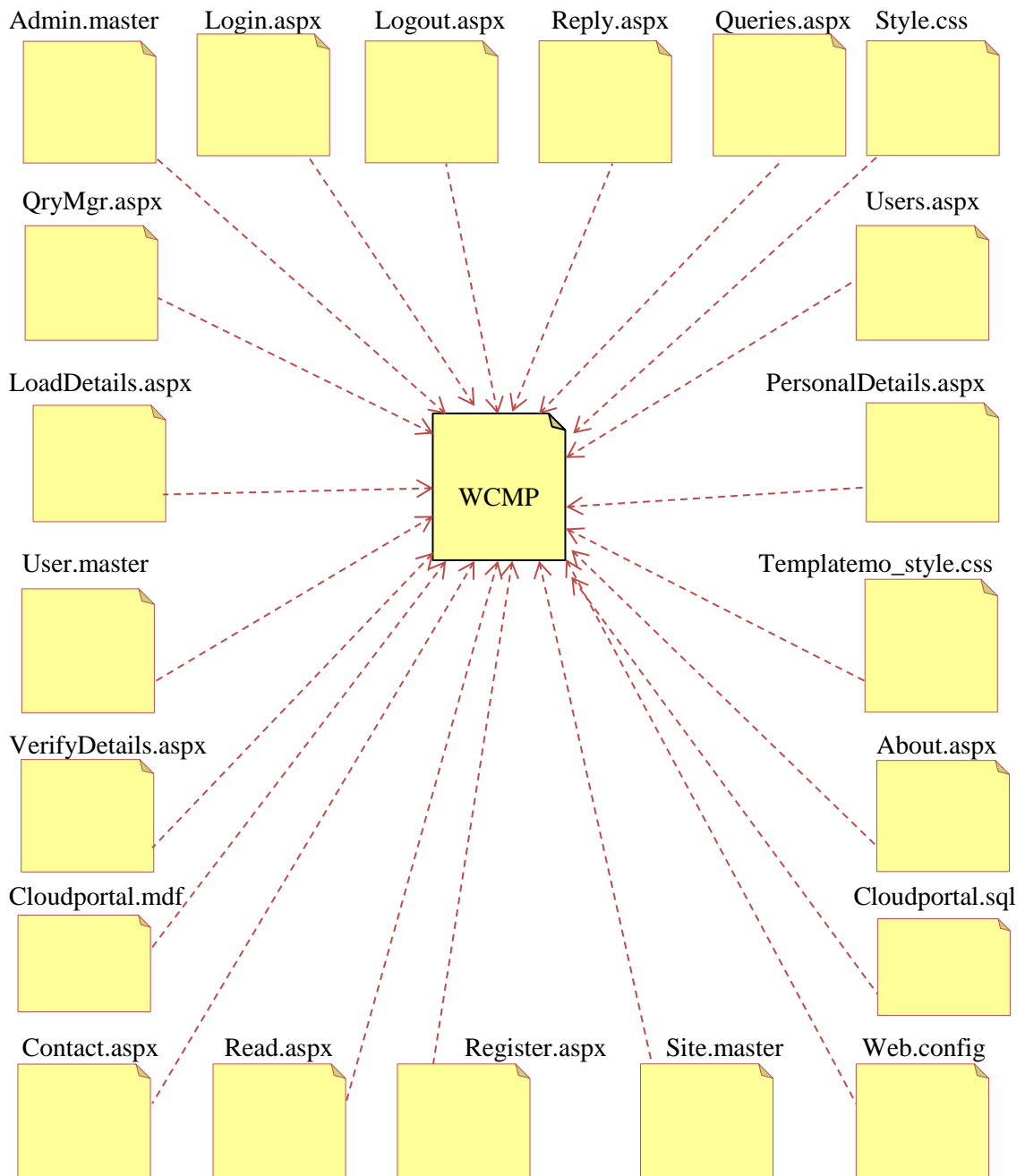


Figure 4.10: Modelling Executable Files and Libraries

viii. Deployment Diagram

A Deployment diagram shows the configuration of run time processing nodes and the components that live on them. With the UML, Deployment diagram is used to visualize the static aspect of these physical nodes and their relationships and to specify their details for construction, as in Figure 4.11.

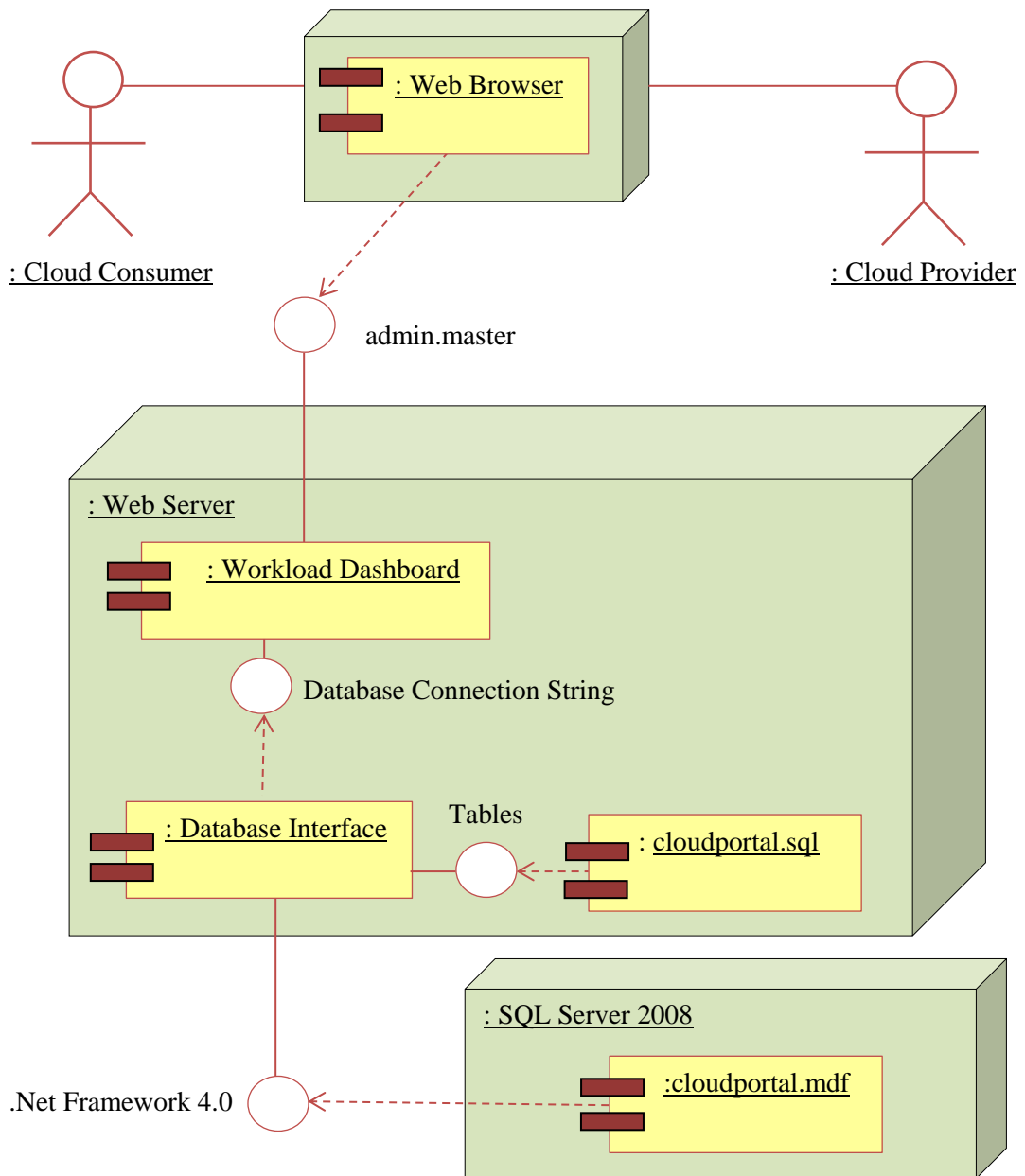


Figure 4.11: Deployment Diagram of CWMP

4.5 Framework Design

Data Flow Diagram (DFD) is used to represent the design description of CWMP.

4.5.1 Design of CWMP Depicted through DFD

The Cloud workload details are gathered through the Cloud Workload Management Portal (CWMP) from Cloud consumer. Web browser acts as an interface for both consumer and provider. The Cloud provider will generate the workload schedule based on the workloads details specified by the user. The workload generated by Cloud provider is based on the four resource scheduling policies, to allocate the resources to the Cloud workloads efficiently.

Data Flow Diagram (DFD) is an interface between the real world activities and an understanding of how an approach can be converted into computer system. Figure 4.12 shows the context or 0 Level DFD of CWMP. This context-level DFD is next exploded, to produce a Level 1 DFD that shows some of the detail of the CWMP being modelled.

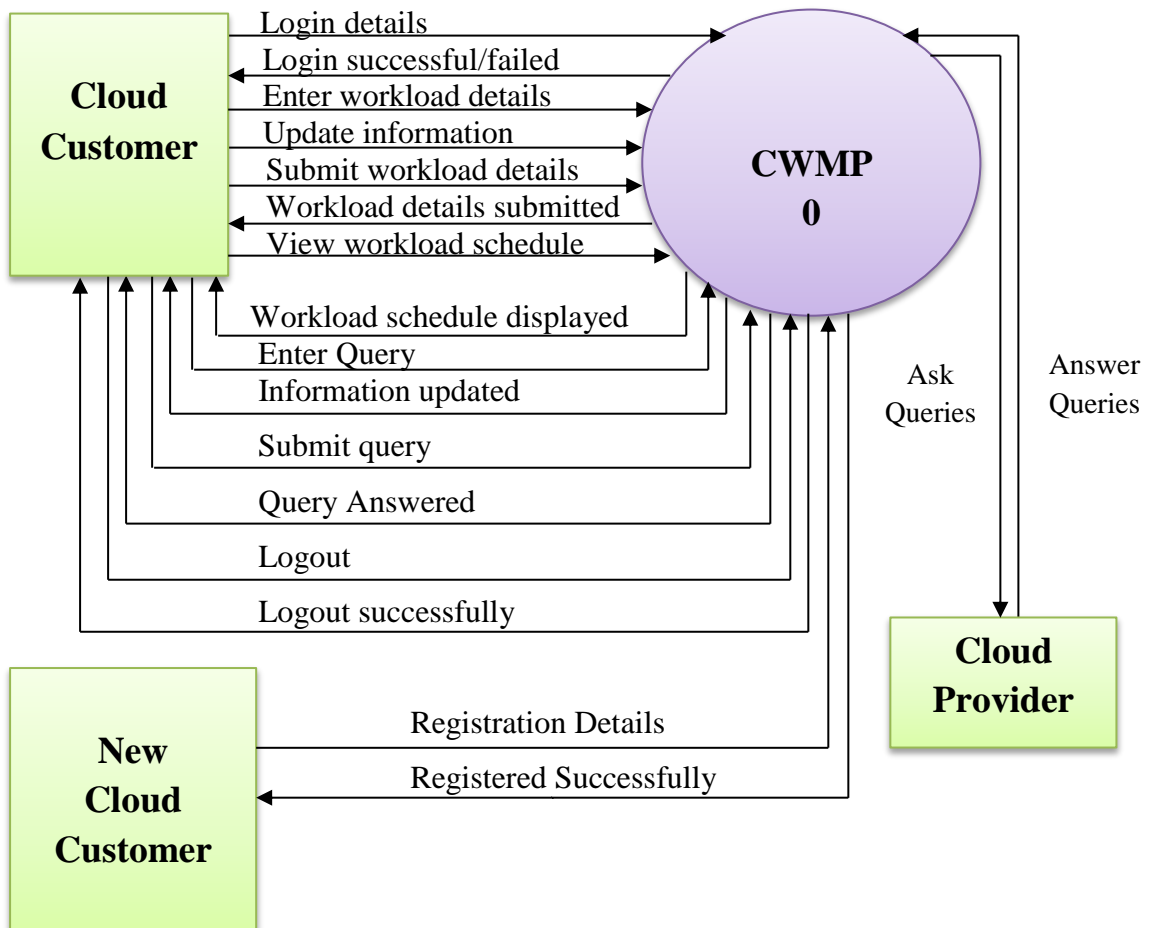


Figure 4.12: Context or 0 Level DFD of the CWMP

The Level 1 DFD shows how the CWMP is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the CWMP as a whole. It also identifies internal data stores that must be present in order for the CWMP to do its job, and shows the flow of data between the various parts of the CWMP. Figure 4.13 shows the Level 1 DFD of the CWMP. There are four types of databases in Cloud Workload Management Portal named; customer database, registration database, workload database and resource database. The workload handler will handle all the incoming Cloud workloads and maps the Cloud workloads to the appropriate resources efficiently based on workload details provided by Cloud provider. Query

handler is used to manage the incoming queries from the user side and forward to Cloud provider. The query will be answered by the Cloud provider.

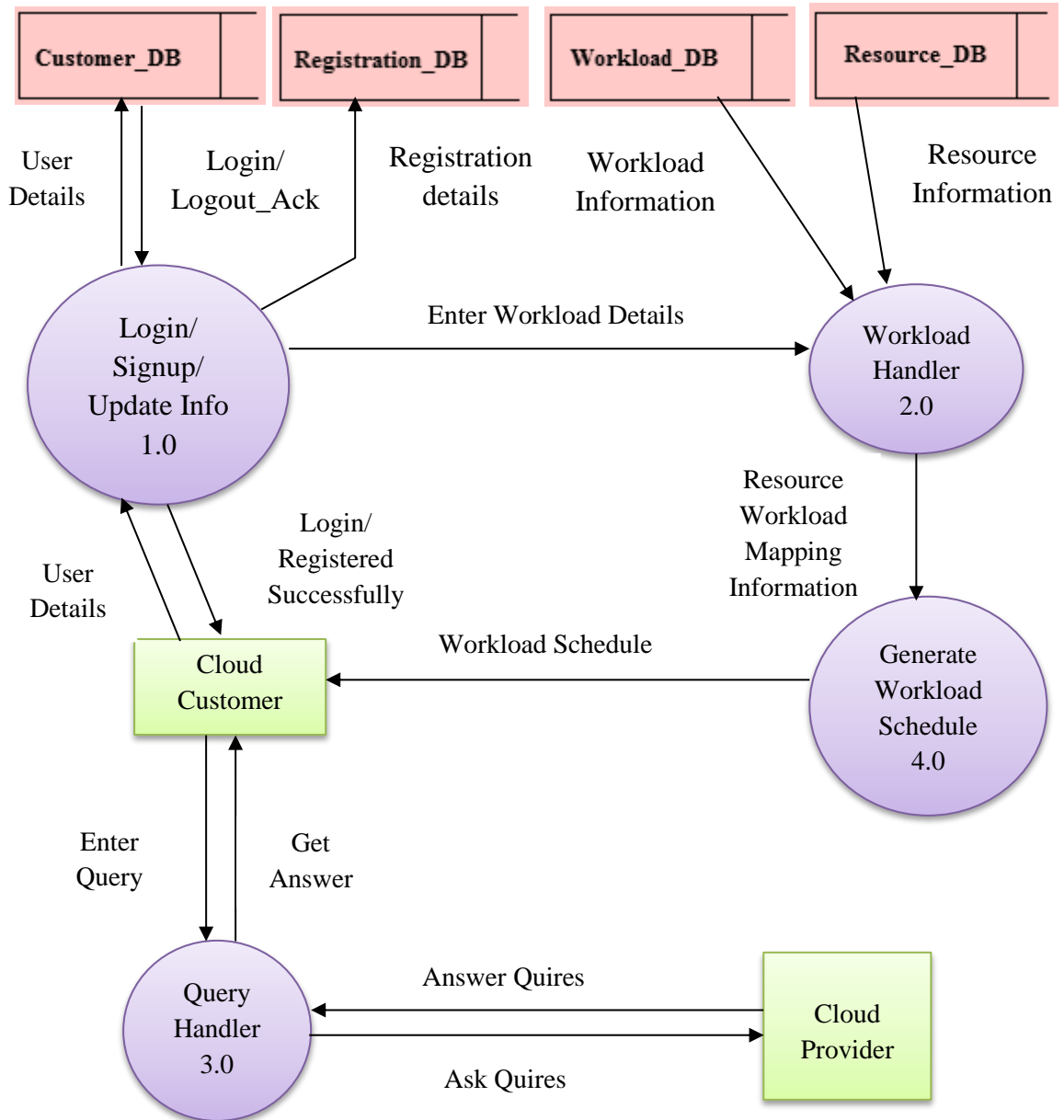


Figure 4.13: Level 1 DFD of the CWMP

4.6 Workload Identification and Analysis

In this section, the Cloud workloads identified and analysed based on the different key Quality of Service (QoS) requirements.

4.6.1 Workload Identification and Description

The types of workload that has been identified during workload analysis, these are Websites, Technological Computing, Endeavour Software, Performance Testing, Online Transaction Processing, E-Com, Central Financial Services, Storage and Backup Services, Production Applications, Software/Project Development and Testing, Graphics Oriented, Critical Internet Applications and Mobile Computing Services [31] [33] [35] [36] [41] [99] [100]. The identified Cloud workloads are described below:

- Websites - Free available websites for social interaction and for large number of Cloud consumers there are informational websites. The quality attributes for this workload are large amounts of reliable storage, high network bandwidth, performance and high availability.
- Technological Computing - It includes bioinformatics, atmospheric modelling, and other numerical computation. The quality attribute for this workload is computing capacity.
- Endeavour Software - It includes email servers, SAP, enterprise content management. The quality attributes for this workload are security, high availability, customer confidence level and correctness.
- Performance Testing - It includes simulation of large workloads to test the performance characteristics of software under development. The quality attribute for this workload is computing capacity and performance.
- Online Transaction Processing - It includes online insurance policies and online banking. The quality attributes for this workload are security, high availability, internet accessibility and usability.
- E-Com - It includes super marketing. The quality attributes for this workload are variable computing load and customizability.
- Central Financial Services - It includes banking and insurance systems. The quality attributes for this workload are security, high availability, changeability and Integrity.
- Storage and Backup Services - It includes general data storage and backup. The quality attributes for this workload are reliability and persistence.

- Productivity Applications - It includes users signing up for mails, word editors etc. The quality attributes for this workload are network bandwidth and latency, data backup and security.
- Software/Project Development and Testing - It includes software development of web applications with Rational Software Architect, Microsoft Visual Studio etc. The quality attributes for this workload are user self-service rate, flexibility, creative group of infrastructure services and testing time.
- Graphics Oriented - It includes animation and visualization software applications. The quality attributes for this workload are network bandwidth and latency, data backup and visibility.
- Critical Internet Applications - It includes web applications including huge amount of scripting languages. The quality attributes for this workload are serviceability, high availability and usability.
- Mobile Computing Services - It includes Servers to support rich mobile applications. The quality attributes for this workload are portability, high availability and reliability.

4.6.2 Workload Analysis

A distinct workload (or a whole application) used by a set of consumers and a smaller facility may be used in dissimilar environments. The different applications have different set of requirements and characteristics. Some Clouds are natural fits for certain classes of workloads (i.e. Web Applications) whereas for another type of workloads (i.e. Batch), other Cloud services (AWS) are more necessary [1]. The aim of workload analysis is to look at different aspects or characteristics of an enterprise application to determine the feasibility of moving or porting the application in the Cloud. This analysis also provides input to execution method, Cloud service choice and a preliminary business worth valuation (Cost benefit analysis) [12] [101] [102] [103].

- a) Cloud Workload: The Cloud workload is an abstraction of work of that instance or set of instances going to perform [24]. For Example: Running a web services or being a Hadoop data node are valid workloads. A workload is a self-governing services or group of code that can be implemented; workload does not depend on outside demands [27]. A workload can be minor or whole

application [104]. Industrial communities have to energetically handle workloads so to check how applications are running. The abstraction is a technique to retain the procedural information away from the consumer. The outcome of this abstraction is a sort of service that makes it easier to have a distinct function with a defined determination [31]. The services live in a container with an Application Programming Interface (API) so it can be simply relocated from one place to another [105]. The term workload in the context of Cloud Computing is an abstraction of the use to which Cloud consumers put their Virtual Machines on the Cloud environment. For example, on desktop sessions a desktop workload might be supporting a number of Cloud consumers logging. To support an enterprise's SAP system through a system of Virtual Machines working together [106]. Workloads are an important distinctive distinguishing the requirements for Cloud Computing [33] [107] [108].

b) Workload Constraints - The following are some constraints with respect to Cloud workloads:

- i. A workload may be time bound ("run for 1 hours") or time unbound.
- ii. A workload may have a particular begin time or elastic begin time.
- iii. A workload may have hard stop time (for example: must finish by a certain time in the future).
- iv. A workload can be interruptible or must run without interruptions.
- v. A workload may have a certain lower limit of the resource that it needs (workload different but applications runs inside the workload are same).
- vi. A workload may have urgency associated with it.
- vii. A workload may have budget associated with it.

c) Workload Assumptions - The assumptions of Cloud workloads is described below [109]:

- i. All workloads are aperiodic.
- ii. Deadline and best-effort workloads are splittable.
- iii. All workloads are independent and the scheduling algorithm assumes that there is no communication among workloads.
- iv. Scheduler accepts a new workload only when it can prepare a feasible schedule for all the old workloads, which were already accommodated in Scheduler as well as new workload.

- d) Workload Cost - It includes the hardware cost, software cost, application maintenance and provision charges etc.
- e) Workload Characteristics - The following are the characteristics of the Cloud workloads:
 - i. Unstable Demand - When a workload has an unchanging and sporadic demand, having dictated and well sized structure for that workload is possibly extra effective than reimbursing hourly charges for VMs in a public Cloud or building and using a private and automatic Cloud.
 - ii. Standard - Usefulness in Cloud Computing is realized appreciations to virtualization and automation. Automation is budget effective if there is a lesser set of features (in SaaS services) or sections of software (in PaaS or IaaS services) present in the directory.
 - iii. Self-governing - If a workload needs heavy communication with another system, relocations of that workload only to a public Cloud situation might affect performance undesirably because of concerns with latency and bandwidth between the data centre and public Cloud situation. Though bandwidth can every time be increased, latency is harder to decrease lower than a lowest threshold unless 1 and 0 can travel faster than light.
 - iv. Not Critical - The workloads with extraordinary challenging necessities (for example availability, response time, recovery time, recovery point and security) might not be organized to be presented in public Clouds so far. Service stages offered by public Clouds did not frequently fulfil the necessities of critical workloads.
- f) Workload Categorization: Distinctive computing workloads involve four basic parts. Practically all applications have their four parts but mostly not stable. Workload based on different applications has been summarized in Table 4.1.
 - i. CPU Oriented Workloads - These applications contain scientific calculation with important data munching, encryption and decryption, compression and decompression and so on.
 - ii. Memory Oriented Workloads - These applications contain in memory caching servers, in memory data servers.
 - iii. Networking Oriented Workloads - These applications are web servers and network load balancers.

- iv. Storage Oriented Workloads - These applications include file serving and data mining.

Table 4.1: Workloads Based on Different Applications

Consumer Applications	Wrapped Applications
Developed and maintained by an enterprise.	Respective vendor is in control of its execution, wrapping, and delivery.
The organization has authority over its design and choosing technology.	Maintained pattern and product design.

- g) Workload Rate - A workload may be used at the last days of the month or after every fixed time (periodically), workload occurrence is important for cost-benefits analysis.
- h) Workload Execution Modes - The workload can be implemented in real time (i.e. online) as well as implemented at every time in a batch mode summarized in Table 4.2.

Table 4.2: Workloads Based on Resource Requirement and Programming Model

Criteria	Batch	Online
Resource Requirement	Require specific capacity in terms of storage and compute resources (CPU, Memory) to finish the job in a timely manner.	The network bandwidth may be more critical.
Programming Model	Some batch jobs may be implemented over a framework like Hadoop.	For online workloads a PaaS may be the best option.

- i) Workload as Independent Objects - A workload has no dependencies. It's a distinct set of application logic that can be implemented individually of a particular application.

4.6.3 Cloud Workload Classification

After identification and analysis of workloads, they are classified on the basis of specific features. Different workloads have different features in terms of security needs, network needs, variability of load, back-up services, network bandwidth needs,

computing capacity and other QoS metrics [112]. Table 4.3 summarizes the classifications of Cloud workloads.

Table 4.3: Classification of Cloud Workloads

Group	Workloads
Server Oriented	Websites, Technological Computing, Endeavour Software, Performance Testing, Online Transaction Processing, E-Com, Central Financial Services, Storage and Backup Services.
Client Oriented	Production Applications, Software/Project Development and Testing, Graphics Oriented, Critical Internet Applications.
Mobile Oriented	Mobile Computing Services

The High level workloads as shown in Figure 4.14 based on different classification groups. The new workloads made possible for Cloud are collaborative care, medical imaging, financial risk and energy management.

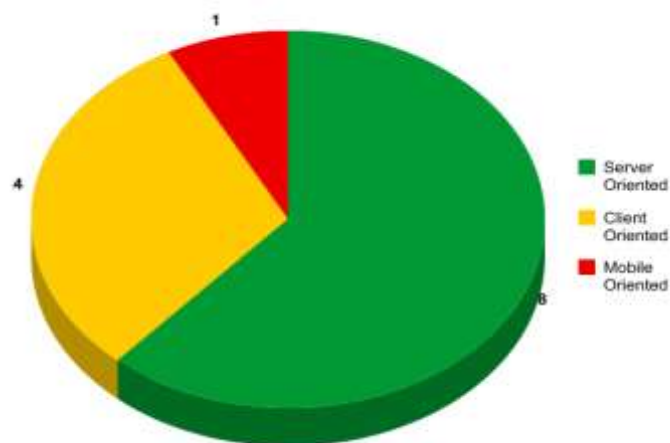


Figure 4.14: High Level Workload Groups

From the literature, the existed metrics have been identified that can be applied to Cloud workloads to measure Network Bandwidth, Integrity, Usability, Reliability, Availability, Changeability, Latency, Cloud Customer confidence Level, Variable Computing Load, Customizability, Testing time, User Self Service Rate, Reliable Storage, Database Backup, Correctness, Service Visibility, Serviceability, Computing Capacity, Flexibility, Internet Accessibility, Persistence, Portability, Security and Performance of Cloud services [101] [102] [110] [111] [112] [113] [114] [115] [116] [117] [118].

4.7 Cloud Workload Based K-Means Algorithm for Clustering of Workloads

To schedule the resources efficiently, the clustering of Cloud workloads will be done. For clustering of Cloud workload, assign weights to different quality attributes based on the importance for particular Cloud workload.

4.7.1 Assign Weights to Quality Attributes

Though, several researchers suggested models and approaches to make quality evaluation i.e. quantitative and less subjective so far quality attributes weighting is totally subjective, depending on domain experts' capabilities to directly weight the attributes. Awkwardly, understanding the significance and subsequently the weights is not simple. It is inferred by several researchers that using ranks to produce weights by various well-known formulas is more trustworthy than just directly assigning weights to attributes. This is for the reason that generally even professionals and decision makers are more self-confident about the ranks of selected attributes than their weights, and they can approve on ranks more easily. In this thesis, the data (average of weights) collected from existing research papers from reputed journals are used because the researchers assigns weights to quality attributes with respect to context in which that quality attribute is used. The relation between Quality Attributes and Cloud Workloads has been described in the form of Matrix as shown in Table 4.4. The range of weight scale has been assumed from 1 (Minimum) to 5 (Maximum). The weights are assigned according to the importance of a requirement for a particular Cloud workload. If any quality attribute is not important for a particular Cloud workload then zero or NA (Not Available) is assigned. These attributes are almost the same according to the international research. Through the questionnaire, the weights for various quality attributes can be assigned from different types of research papers as shown in Table 4.5. Later receiving the required information the average of every attribute has been taken and that average is the approximate weight (percentage) of that quality attributes [119]. The consequence of collected data is used by the following formula to calculate quality attributes weight:

$$W(i, j) = \frac{1}{N_f \times M_v} \times \sum_{k=1}^{N_f} R_k \times 100$$

Where in $W(i, j)$, i is cloud workload and j is quality attribute of that workload, N_f is number of research papers used to collect data, M_v is maximum value for a quality attribute and R_k is response for an attribute; the value of $W(i, j)$ will be in the range 0% - 100%. An analysis has been conducted to acquire the data from 15 research papers of Cloud Computing from reputed journals about Cloud workloads with the objective to know that how to assign the weights to the quality attributes according to significance [120] [121] [122] [123] [124] [125] [126] [127] [128] [129] [130] [131] [132] [133] [134]. After getting the responses, an industry standard baseline and acceptable weights to the quality attributes has been defined.

Table 4.4: Quality Attributes and Cloud Workloads Matrix

Quality Attributes	w1	w2	w3	w4	w5	w6	w7	w8	w9	w10	w11	w12	w13
Reliable Storage	√												
Latency									√		√		
Computing Capacity		√		√									
Data Backup									√		√		
Customer Confidence Level			√										
User Self-Service Rate										√			
Correctness			√										
Flexibility										√			
Security			√		√		√		√				
Internet Accessibility					√								
Creative Group of Infrastructure Services										√			
Usability					√							√	
Variable Computing Load						√							
Customizability						√							
Testing Time										√			
Changeability							√						
Visibility											√		
Integrity							√						
Reliability								√	√				√
Serviceability												√	
Persistence								√	√				
Portability													√
Performance				√									
High Network Bandwidth	√									√	√		
High Availability	√		√		√		√					√	√

Table 4.5: Questionnaire for Assigning the Weights to Quality Attributes

Question Element	Importance Degree					
1. Websites	NA	1	2	3	4	5
1. Reliable Storage						
2. High Network Bandwidth						
3. High Availability						
2. Technological Computing	NA	1	2	3	4	5
1. Computing Capacity						
3. Endeavour Software	NA	1	2	3	4	5
1. Security						
2. High Availability						
3. Customer Confidence Level						
4. Correctness						
4. Performance Testing	NA	1	2	3	4	5
1. Computing Capacity						
2. Performance						
5. Online Transaction Processing	NA	1	2	3	4	5
1. Security						
2. High Availability						
3. Internet Accessibility						
4. Usability						
6. E-Com	NA	1	2	3	4	5
5. Variable Computing Load						
6. Customizability						
7. Central Financial Services	NA	1	2	3	4	5
1. Security						
2. High Availability						
3. Changeability						
4. Integrity						
8. Storage and Backup Services	NA	1	2	3	4	5
1. Reliability						
2. Persistence						
9. Productivity Applications	NA	1	2	3	4	5
1. Network Bandwidth						
2. Latency						
3. Data Backup						
4. Security						
10. Software/Project Development and Testing	NA	1	2	3	4	5
1. User Self-Service Rate						
2. Flexibility						
3. Rich Set of Infrastructure Services						
4. Testing Time						
11. Graphics Oriented	NA	1	2	3	4	5
1. Network Bandwidth						
2. Latency						
3. Data Backup						
4. Visibility						
12. Critical Internet Applications	NA	1	2	3	4	5
1. High Availability						
2. Serviceability						
3. Usability						
13. Mobile Computing Services	NA	1	2	3	4	5
1. High Availability						
2. Reliability						
3. Portability						

The conversion metric is used to assign the values (minimum = 1 and maximum = 5) corresponding to the percentage as shown in Table 4.6.

Table 4.6: Conversion Metric

Approximate Weight (%)	Weight
0-20	1
20-40	2
40-60	3
60-80	4
80-100	5

Research papers have been selected randomly to gather the data through data collection technique. Likert scale is oscillating from 1 [lowest] to 5 [highest] to gather the data [135]. All quality attributes to make it convenient for the other researchers to assign the values has been identified. The objectives of the data collection are to 1) to minimize the variations in limits of data, 2) Gather the information about the level of quality attribute, 3) Gather the information about the pervious weights assigned, 4) Gather the information about the researcher (Experience, research successfully completed etc.). This information determines the authenticity of the data that is received from different research papers. The result of the data analysis is as follows; total 15 research papers of different contexts have been studied and maximum possible value for an attribute is 5. For example calculating the average of “Usability” attribute is as following:

$N_f = 12$, $i =$ Online Transaction Processing and $j =$ Usability, $M_v = 5$ and sum of the responses $\sum_{k=1}^{12} R_k = 32$

$$W(i, j) = \frac{1}{12 \times 5} \times 32 \times 100 = 53.33$$

For $W(i, j) = 53.33$, the average weight is assigned for usability is 3. Through this the average weights for every quality attribute has been calculated.

4.7.2 Cloud Workload Based K-Means Algorithm for Clustering of Workloads

As workload demands vary widely and is quite fluctuating simple static resource provisioning results in over and under provisioning. Cloud Workload K-Means Clustering Algorithm is non-hierarchical method initially takes the number of workloads of the population equal to the final necessary number of clusters. The

actual essential number of clusters is selected such that the workloads are mutually farthest apart based on key QoS requirements in this step. Next, it examines each workload in the population and assigns it to one of the clusters depending on the minimum distance. The cluster centroid's (C) position is recalculated every time a workload is added to the cluster and this continues until all the workloads are grouped into the final required number of clusters (C_t) [136]. The algorithm used for clustering of Cloud workload as shown in Figure 4.15. Cloud Workload K-Means Clustering Algorithm calculate the distance of between each workloads and select that pair which show the minimum distance and remove it from actual Workload Set (W). Then took one workload from Workload Set ($W = \{W_1 W_2 \dots \dots \dots \dots W_t\}$) and calculate the distance between selected workload and standard workload from Workload Set (W) and add with that cluster which show the minimum distance. Repeat this process till threshold value achieved. If number of value is less than k then again calculate the distance between each workload from the rest Workload Set (W) and repeat that process till k cluster formed.

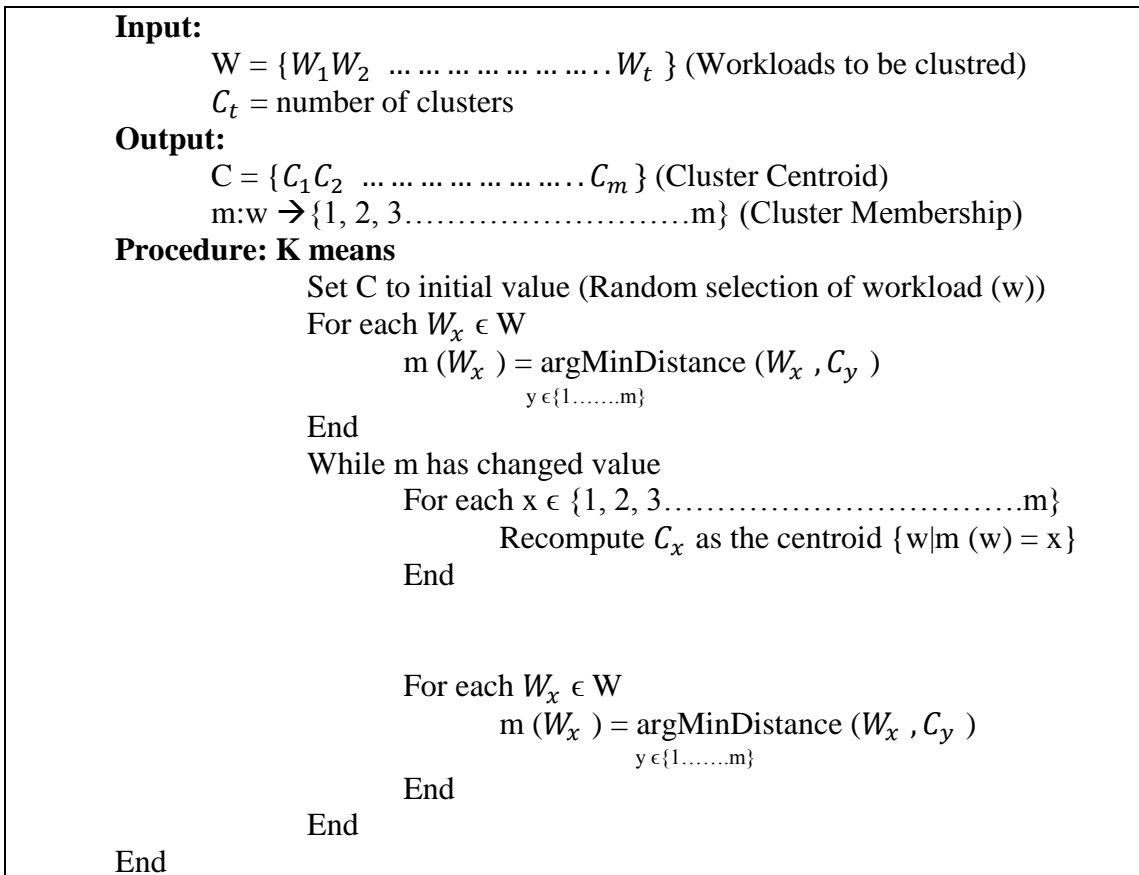


Figure 4.15: K-Means Algorithm for Clustering of Cloud Workloads

4.7.3 Weight Assignment for Cloud Workloads

The range of weight scale has been assumed from 1 (Minimum) to 5 (Maximum). The weights are assigned according to the importance of a requirement for a particular Cloud workload as shown in Table 4.7.

Table 4.7: Workloads with their Requirements and Weights

Id	Workload	Requirements	Weights Assigned
W1	Web sites	<ul style="list-style-type: none"> ➤ Reliable storage ➤ High network bandwidth ➤ High availability 	3 3 5
W2	Technological Computing	<ul style="list-style-type: none"> ➤ Computing capacity 	5
W3	Endeavour Software	<ul style="list-style-type: none"> ➤ Security ➤ High availability ➤ Customer confidence Level ➤ Correctness 	5 5 3 3
W4	Performance Testing	<ul style="list-style-type: none"> ➤ Computing capacity 	5
W5	Online Transaction Processing	<ul style="list-style-type: none"> ➤ Security ➤ High availability, ➤ Internet accessibility ➤ Usability 	5 3 5 3
W6	E-Com	<ul style="list-style-type: none"> ➤ Variable computing load ➤ Customizability 	5 3
W7	Central Financial Services	<ul style="list-style-type: none"> ➤ Security ➤ High availability ➤ Changeability ➤ Integrity 	5 3 1 5
W8	Storage and Backup Services	<ul style="list-style-type: none"> ➤ Reliability ➤ Persistence 	5 3
W9	Productivity Applications	<ul style="list-style-type: none"> ➤ Network bandwidth ➤ Latency ➤ Data backup ➤ Security 	2 3 4 5
W10	Software/Project Development and Testing	<ul style="list-style-type: none"> ➤ User self-service rate ➤ Flexibility ➤ Creative group of infrastructure services ➤ Testing time 	4 4 1 5
W11	Graphics Oriented	<ul style="list-style-type: none"> ➤ Network bandwidth ➤ Latency ➤ Data backup ➤ Visibility 	3 3 5 4
W12	Critical Internet Applications	<ul style="list-style-type: none"> ➤ High availability ➤ Serviceability ➤ Usability 	5 4 3
W13	Mobile Computing Services	<ul style="list-style-type: none"> ➤ High availability ➤ Reliability ➤ Portability 	3 5 2

Abbreviations for the workload and their corresponding requirements have been presented in Table 4.8.

Table 4.8: Abbreviations of Workload Requirements

Network Bandwidth	R1	Variable Computing Load	R13
Integrity	R2	User Self Service Rate	R14
Security	R3	Reliable Storage	R15
Usability	R4	Database Backup	R16
Reliability	R5	Correctness	R17
Availability	R6	Visibility	R18
Changeability	R7	Serviceability	R19
Latency	R8	Computing Capacity	R20
Customer confidence Level	R9	Flexibility	R21
Portability	R10	Internet Accessibility	R22
Customizability	R11	Persistence	R23
Testing time	R12	Creative group of infrastructure services	R24

Based on the importance of a requirement for a particular Cloud workload the data values have been assigned as shown in Table 4.9.

Table 4.9: Data Values for Workloads

Workloads	Requirements	Value 1	Value 2	Value 3	Value 4
W1	R15, R1, R6	3	3	5	0
W2	R20	5	0	0	0
W3	R3, R6, R9, R17	5	5	3	3
W4	R20	5	0	0	0
W5	R3, R6, R22, R4	5	3	5	3
W6	R13, R11	5	3	0	0
W7	R3, R6, R7, R2	5	3	1	5
W8	R5, R23	5	3	0	0
W9	R16, R1, R3, R8	2	3	4	5
W10	R14, R21, R24, R12	4	4	1	5
W11	R1, R8, R16, R18	3	3	5	4
W12	R6, R19, R4	5	4	3	0
W13	R5, R6, R10	3	5	2	0

Let the four seeds [137] be the four workloads as shown in Table 4.10.

Table 4.10: The Four Seeds for Given Workloads

Seed	Value 1 (V1)	Value 2 (V2)	Value 3 (V3)	Value 4 (V4)
s1	5	0	0	0
s2	5	3	0	0
s3	3	3	5	0
s4	5	3	5	3

Now compute the distance using the four values (Weights Assigned) and using the sum of differences for simplicity (i.e. using the K-median method [138]). The distance values for all the Cloud workloads are given in Table 4.11, wherein columns 6, 7, 8 and 9 give the four distances from four seeds respectively. Based on these distances workload is allocated to the nearest cluster [139] and the result of first iteration as shown in Table 4.11.

Table 4.11: First Iteration- Allocating Each Cloud Workload to the Nearest Cluster

C1	5	0	0	0	Distance From Clusters				Allocation to the nearest Cluster
					Distance From C1	Distance From C2	Distance From C3	Distance From C4	
C2	5	3	0	0					
C3	3	3	5	0					
C4	5	3	5	3					
Workload	V1	V2	V3	V4					
W1	3	3	5	0	6	3	0	5	C3
W2	5	0	0	0	0	3	6	11	C1
W3	5	5	3	3	11	8	5	0	C4
W4	5	0	0	0	0	3	6	11	C1
W5	5	3	5	3	11	8	5	0	C4
W6	5	3	0	0	3	0	3	8	C2
W7	5	3	1	5	9	6	3	2	C4
W8	5	3	0	0	3	0	3	8	C2
W9	2	3	4	5	9	6	3	2	C4
W10	4	4	1	5	9	6	3	2	C4
W11	3	3	5	4	10	7	4	1	C4
W12	5	4	3	0	7	4	1	4	C3
W13	3	5	2	0	5	2	1	5	C3

First iteration leads to two each workload in first and second cluster, three in third cluster and six in fourth cluster. Table 4.12 compares [140] the cluster means of cluster found in Table 4.11 with the original seeds (s1, s2, s3, s4).

Table 4.12: Comparing New Centroids and the Seeds

	Value 1	Value 2	Value 3	Value 4
C1	5	0	0	0
C2	5	3	0	0
C3	3.6	4	3.3	0
C4	4	3.5	3.1	4.1
s1	5	0	0	0
s2	5	3	0	0
s3	3	3	5	0
s4	5	3	5	3

Use the new cluster means to recomputed the distance of each object to each of the means, again allocating each Cloud workload to the adjacent cluster. Table 4.13 shows the second iteration.

Table 4.13: Second Iteration- Allocating Each Cloud Workload to the Nearest Cluster

C1	5	0	0	0	Distance From Clusters				Allocation to the nearest Cluster
C2	5	3	0	0	Distance From C1	Distance From C2	Distance From C3	Distance From C4	
C3	3.6	4	3.3	0					
C4	4	3.5	3.1	4.1					
Workload	V1	V2	V3	V4					
W1	3	3	5	0	6	3	0.1	3.7	C3
W2	5	0	0	0	0	3	5.9	9.7	C1
W3	5	5	3	3	11	8	5.1	1.3	C4
W4	5	0	0	0	0	3	5.9	9.7	C1
W5	5	3	5	3	11	8	5.1	1.3	C4
W6	5	3	0	0	3	0	2.9	6.7	C2
W7	5	3	1	5	9	6	3.1	0.7	C4
W8	5	3	0	0	3	0	2.9	6.7	C2
W9	2	3	4	5	9	6	3.1	0.7	C4
W10	4	4	1	5	9	6	3.1	0.7	C4
W11	3	3	5	4	10	7	4.1	0.3	C4
W12	5	4	3	0	7	4	1.1	2.7	C3
W13	3	5	2	0	5	2	0.1	4.7	C3

The number of workloads in all the four clusters is again same. A more careful look shows that the clusters have not changed at all. The cluster membership [137] is shown in Table 4.14.

Table 4.14: Cluster Membership

Cluster	Cluster Name	Workloads
C1	Compute	W2, W4
C2	Storage	W6, W8
C3	Communication	W1, W12, W13
C4	Administration	W3, W5, W7, W9, W10, W11

The distribution of various Cloud workloads among four clusters is shown in Figure 4.16.

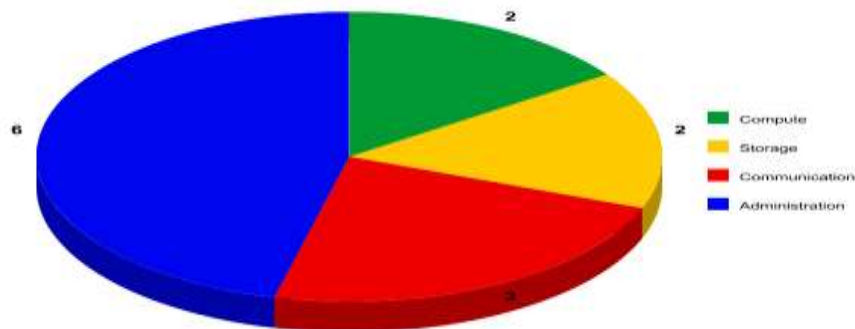


Figure 4.16: Distribution of Various Workloads among Four Clusters

4.7.4 Identified Design Patterns

The Design Patterns identified from the literature [82] [83] [84] [85] [86] [87] [88] [89] [90] [91] [92] are given below:

- a) On-demand Application Instance - It contains the applications that need scale-out and scale-down capabilities. For Example: Let retail store sites be available during special events.
- b) Operative - It includes executing parallel batch jobs or background applications. For Example: Using schedulers for analytics processing by executing background tasks in parallel.
- c) Simple Storage - It deals with storage large amount of unstructured data. For Example: Company storing legal compliance reports in backup store.
- d) Structured Storage - It deals with storing data in a table structure while not demanding full relational semantics. For Example: A structured storage to maintain a Web application's state (shopping basket information)
- e) Service Interface (Web and Web service API) - It contains exposing application capabilities through UI and Web services. For Example: Company building digital asset management solution exposing APIs to other services.
- f) Service-Oriented Integration - It deals with Invocation of an external Web services using Web-standard protocols. For Example: Web applications leveraging Web-hosted live meeting services for collaboration.
- g) Messaging - It contains share messages between applications in a scalable, reliable, and asynchronous way. For Example: Web application informing a scheduler to execute a specific task.
- h) Cloud Deployment - Deploying applications with desired configurations such as scale-out and high-availability requirements. For Example: A retail store configuring Web portal to automatically scale-out when usage exceeds threshold and scale-down as needed.
- i) Design for Operations - How to make my application operations-ready by providing health status and logging. For Example: Developers designing Cloud applications to be operations friendly through GUI.
- j) Service Instance Management - Start, stop, and suspend Cloud apps. Manage service configurations. For Example: A Web application administrator using the service portal to manage application state.

- k) Management Alerts - Sending Immediate Messages, mails, or warnings about resource and billing details. It enables applications to send emails. For Example: Default notification on resource usage.
- l) Service Level Management - Get information on application resource consumption such as processor time, bandwidth. For Example: Looking for billing and resource usage info about the application deployed with billing transparency.

4.7.5 Design Pattern Template

The design pattern template consists of different parts related to all aspects as shown in Figure 4.17.

Design Pattern Name
Requirement
Summary
Problem:
Solution:
Application:
Impact:

Figure 4.17: Design Pattern Template

- Design Pattern Name - The name of the design pattern to be considered.
- Requirement - This is a brief statement that offers the important constraint addressed by the design pattern in the form of a query.
- Summary - The summary table is consisting of statements that jointly offer a brief outline of the design pattern for rapid reference purposes. Moreover, the summary table offers references to interrelated service-orientation design philosophies and service-oriented architectural categories.
- Problem - It is the problem for which the design pattern offers a solution. Problem explanations may also contain common situations that can lead to the problem.

- Solution - This signifies the design solution offered by the design pattern to solve the problem and fulfil the need.
- Application - This section is devoted to defining how the design pattern can be used. It can consist of strategies, implementation details, and occasionally even a suggested process.
- Impact - This section describes significances, budgets and requirements related with the application of a design pattern.

Based on the key characteristics of Cloud workloads, the design pattern are mapped with Cloud workloads. Table 4.15 shows how the Cloud workloads have been distributed among four clusters and their corresponding pattern classification, framework and their description [141]. There are four clusters named Compute, Storage, Communication and Administration.

Table 4.15: Clusters with Patterns and Cloud workloads

Cluster	Cloud Workload	Pattern Classifications	Framework	Description
Compute	Performance Testing	On-demand Application Instance	Applications that need scale-out and scale-down capabilities.	Let retail store sites be available during special events.
	Technological Computing	Operative	Executing parallel batch jobs or background applications.	Using schedulers for analytics processing by executing background tasks in parallel.
Storage	E-Com	Simple Storage	Storing large amount of unstructured data.	Company storing legal compliance reports in backup store.
	Storage and Backup	Structured Storage	Storing data in a table structure while not demanding full relational semantics.	A structured storage to maintain a Web application's state (shopping basket information)
Communication	Website	Service Interface (Web and Web Service API)	Exposing application capabilities through UI and Web services.	Company building digital asset management solution exposing APIs to other services.
	Critical Internet Applications	Service-oriented Integration	Invoking external Web services using Web-standard protocols.	Web applications leveraging Web-hosted live meeting services for collaboration.
	Mobile Computing Services	Messaging	Share messages between applications in a scalable, reliable, and asynchronous way.	Web application informing a scheduler to execute a specific task.

Administration	Endeavour Software and Online financial Services	Cloud Deployment	Deploying applications with desired configurations such as scale-out and high-availability requirements.	A retail store configuring Web portal to automatically scale-out when usage exceeds threshold and scale-down as needed.
	Software/Project Development and Testing	Design for Operations	How to make my application operations-ready by providing health status and logging.	Developers designing Cloud applications to be operations friendly through GUI.
	Central Financial services	Service Instance Management	Start, stop, and suspend Cloud apps. Manage service configurations.	A Web application administrator using the service portal to manage application state.
	Productivity Applications	Management Alerts	Sending Immediate Messages, mails, or warnings about resource and billing detail.	Enabling applications to send emails. Default notification on resource usage.
	Graphic Oriented	Service Level Management	Get info on app resource consumption such as processor time, bandwidth.	Looking for billing and resource usage info about the application deployed with billing transparency.

4.7.6 Cloud Workload Execution Design

In the course of its lifetime, a workload passes through many states as is outlined in Figure 4.18. A workload is an input to the scheduler which allocates it to a set of resources based on its requirements. The workload's status is then changed to SCHEDULED. During the STAGE_IN state, input files and executables required for the workload are staged to the available resource. When this process is completed successfully, then a workload is considered to be SUBMITTED. The workload may be queued while waiting for an available processor and its state changes to PENDING. When the workload starts its execution, it is considered ACTIVE. After the workload has finished executing, it enters the STAGE_OUT stage where its output files are transferred back to the broker. If all its outputs are received and are as expected by the task requirements, then the workload is considered as "DONE". If one of state transitions fails on the available side or the workload has completed on the available side but has not produced the expected result files, then it is considered FAILED and is reset and marked for re-scheduling.

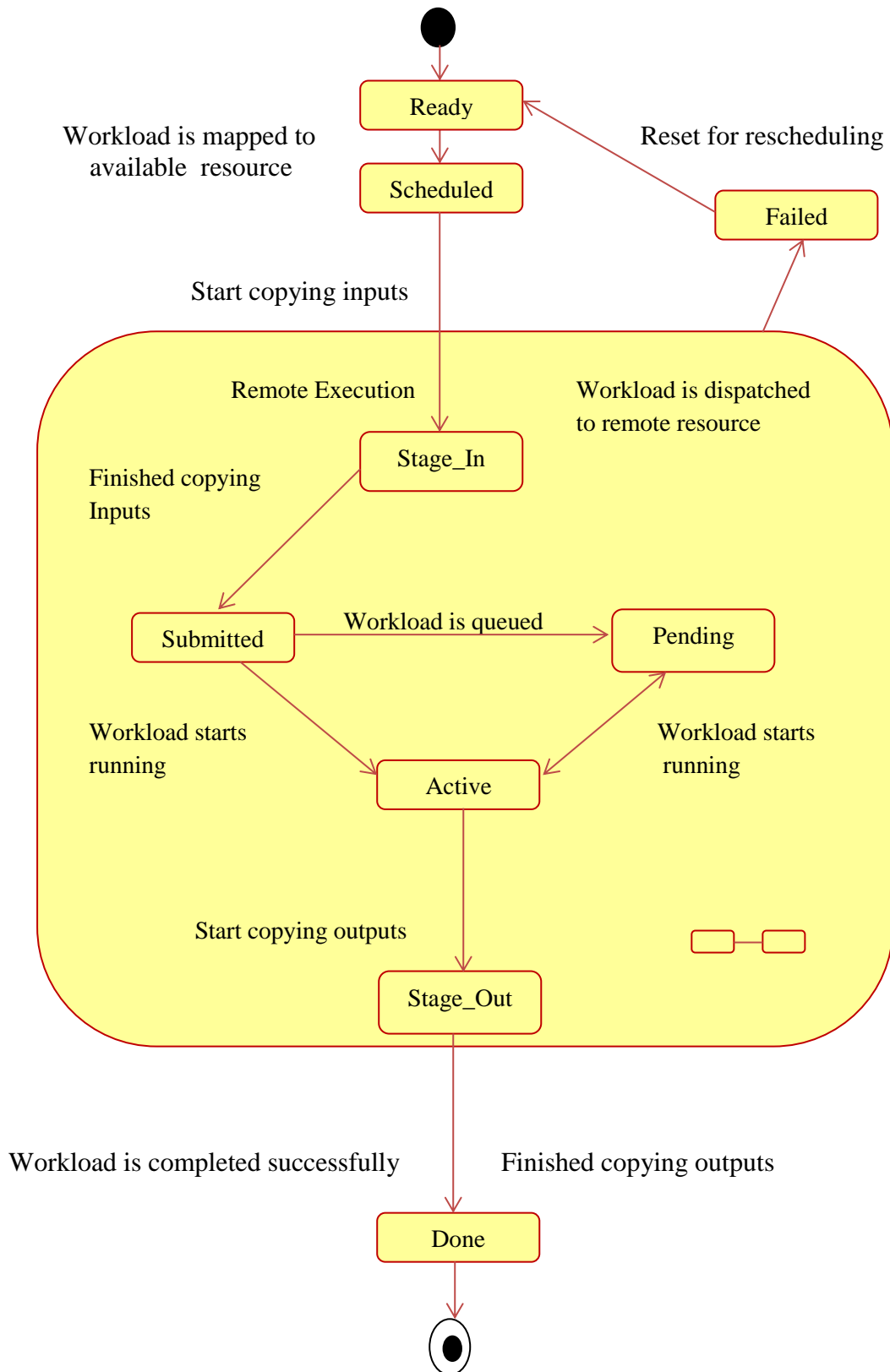


Figure 4.18: State Transition Diagram for Cloud Workload

Cloud Resource Manager manages all the Cloud workloads and resources and map efficiently. There are different entities and interfaces associated with Cloud Resource Manager as shown in Figure 4.19.

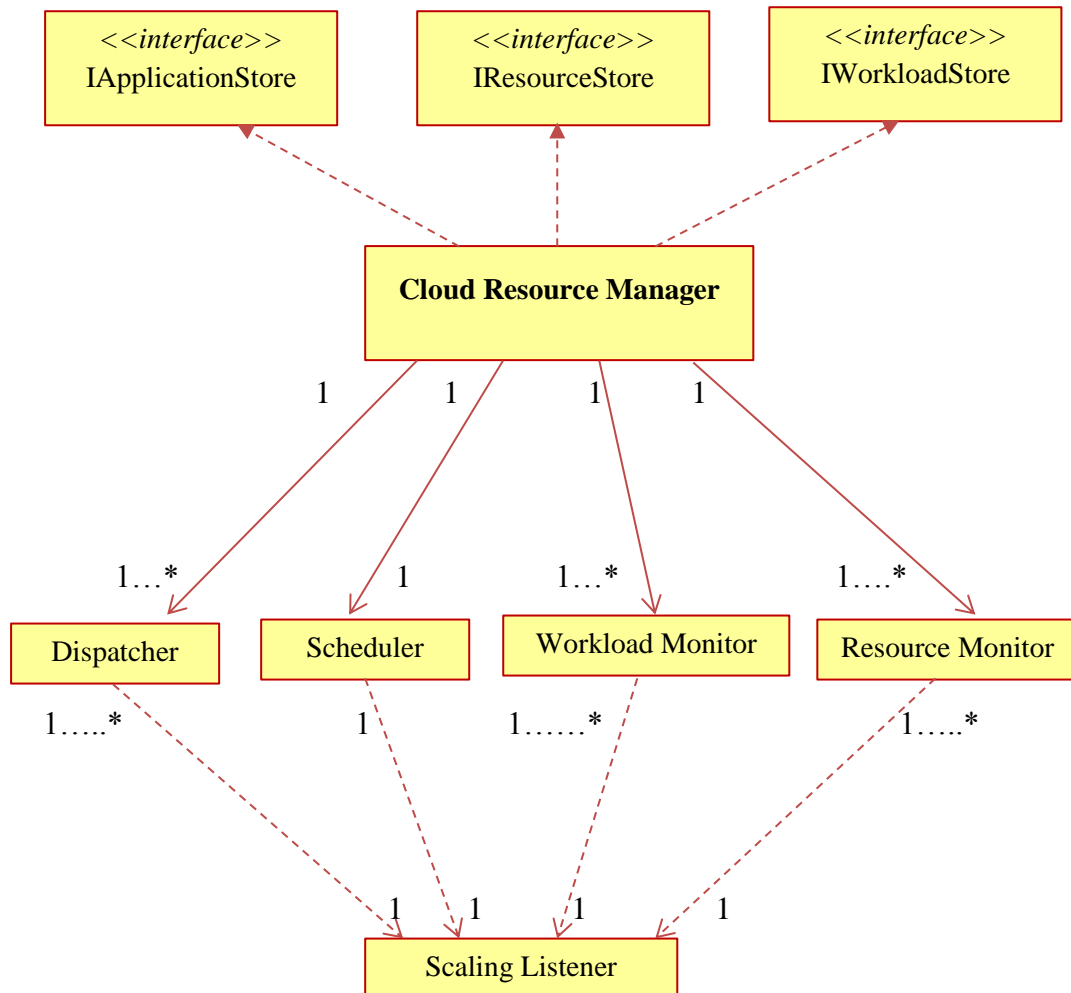


Figure 4.19: Cloud Resource Manager and its Associated Entities

The Scaling Listener is used to handle the request from various Cloud consumers and allocate the Cloud workloads to the available resources efficiently based on the workload details submitted by the Cloud consumer. There are two loops used during resource allocation. First loop is scheduler loop, for computing the resource requirement for a particular Cloud workload. Second loop is discovering processor properties like processing cost, speed and MIPS rating. After execution of every Cloud workload, the processor properties are saved for future purpose. The Sequence Diagram of Resource Monitoring is shown in Figure 4.20.

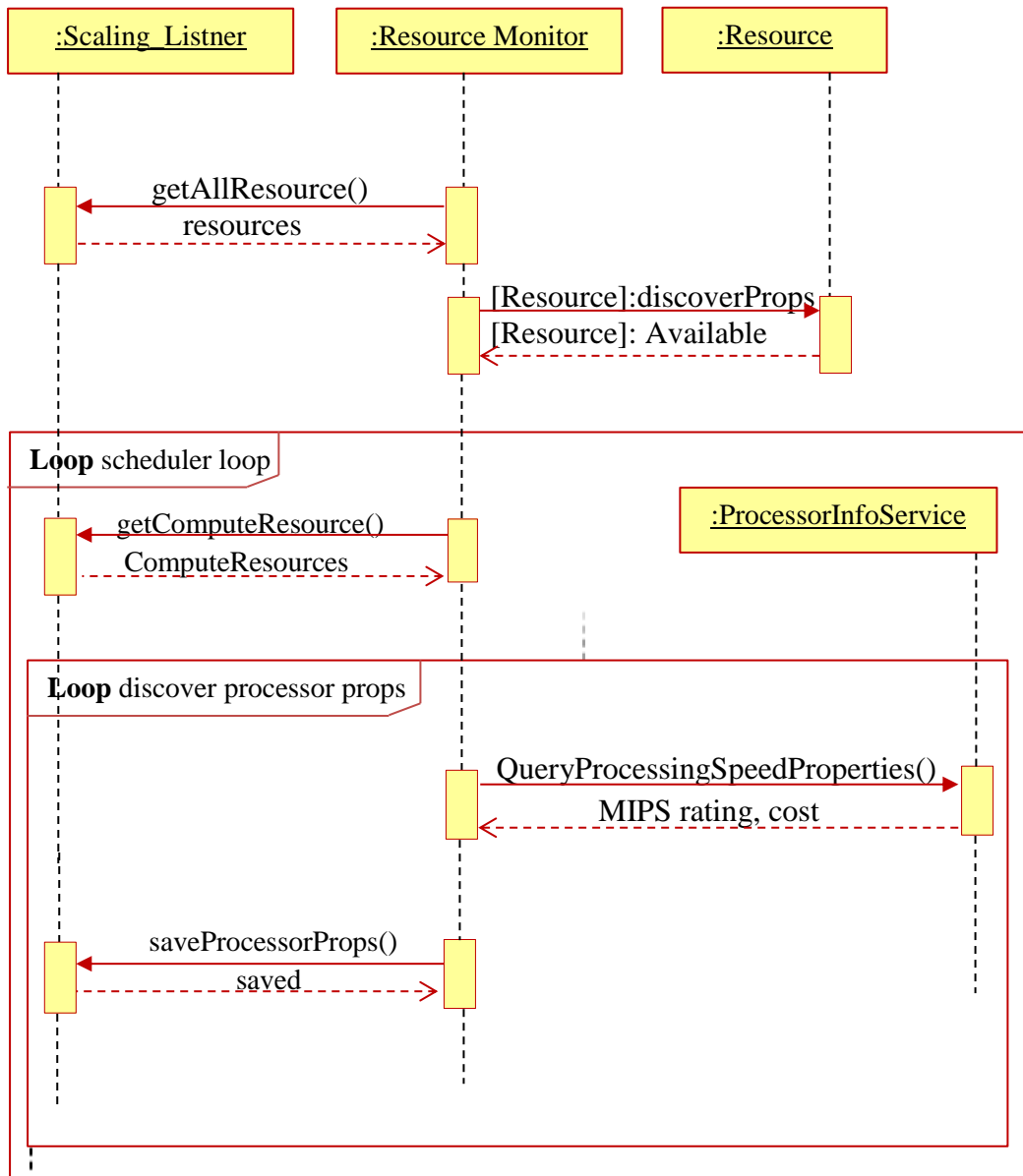


Figure 4.20: Sequence Diagram of Resource Monitoring

The Resource Scheduler schedule the incoming Cloud workloads based on the workloads details. Two loops are used. First loop for scheduling i.e. get Cloud workloads to schedule and second loop is used to ask resources repeatedly and available resources and Cloud workloads mapped efficiently based on the scheduling policies. All the incoming workloads are put into different queue than the queue contains already submitted Cloud workloads. The Sequence diagram of generating Cloud workload schedule is shown in Figure 4.21. The mapping is saved for future purpose.

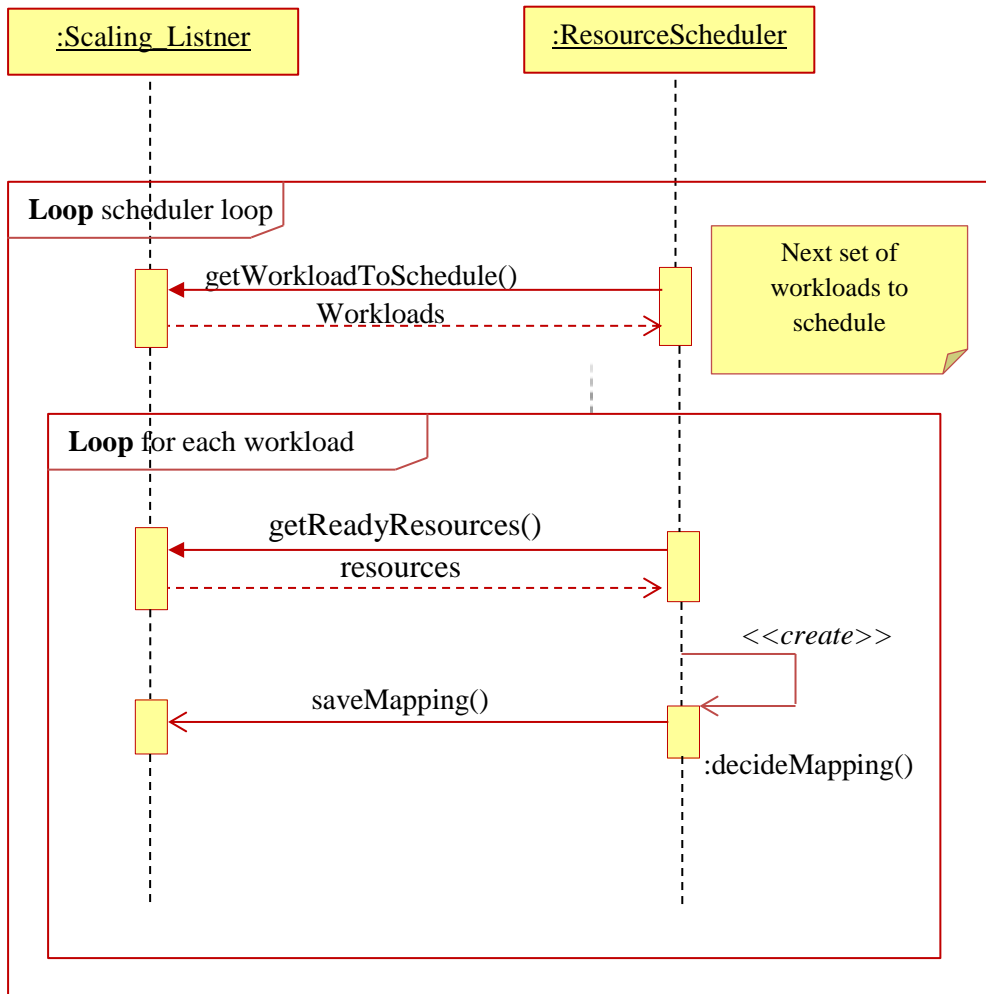


Figure 4.21: Sequence Diagram of Workload Schedule

Dispatcher is used to dispatch the Cloud workloads for execution. One loop is used to dispatch all the Cloud workloads. Scaling Listener gathered Cloud workload details and provided to the dispatcher. A particular Cloud workload is submitted to the compute resource for calculating the amount of resources required. The Workload Wrapper is created by allocating the resources to the Cloud workloads. After that the tentative mapping is committed and stores each mapping. The Sequence Diagram of Dispatching Workloads is shown in Figure 4.22. Workload Monitor is used to check the status of Cloud workloads. The loop workload monitor is used to monitor the incoming Cloud workloads. The queries related to workload monitoring are asked and replied according to the status. After the execution of the Cloud workload, the status is changed by Workload Listener from executing to finish. After the successful execution of Cloud workload the status is updated in the system. The Sequence Diagram of Workload Monitoring is shown in Figure 4.23. There are different clusters of Cloud workloads are created and put the incoming workloads in suitable cluster.

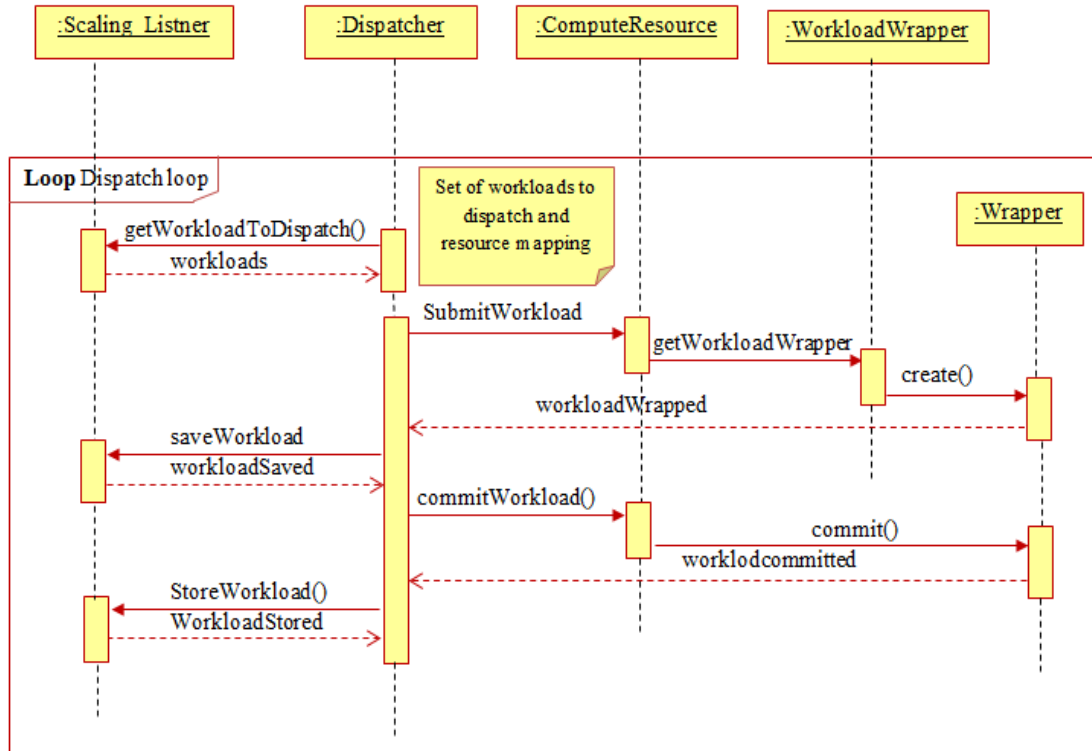


Figure 4.22: Sequence Diagram of Dispatching Workloads

There are eight different classes created to represent the interaction among different entities. Workloads are allocated to the appropriate resources based on the workload description by taking care of QoS. Fitness value for every workload is calculated and analysed. Allocate the resources to Cloud workloads and execute within the defined budget and desired deadline with minimum energy consumption. The Class diagram for Workload Allocation is shown in Figure 4.24.

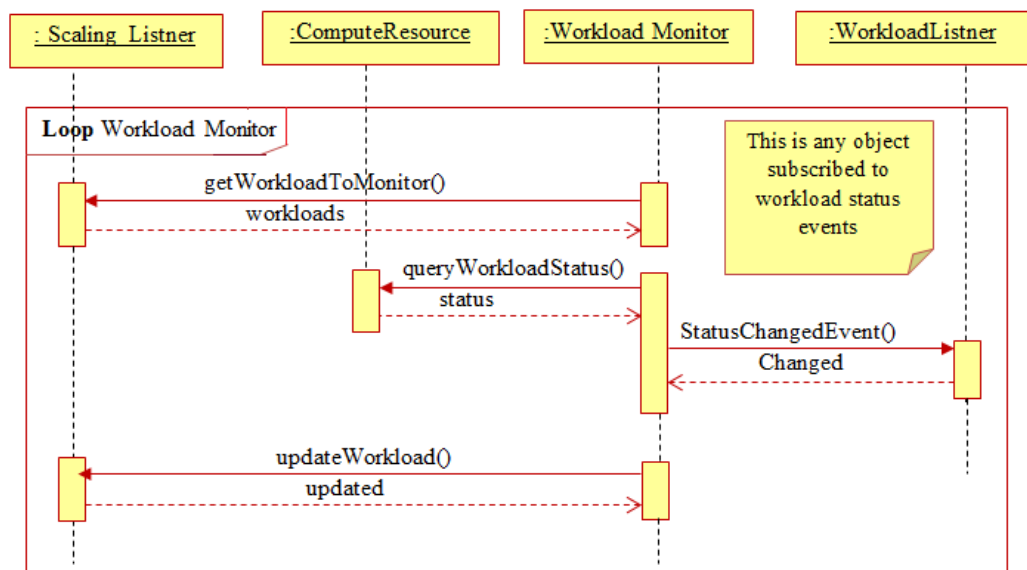


Figure 4.23: Sequence Diagram of Workload Monitoring

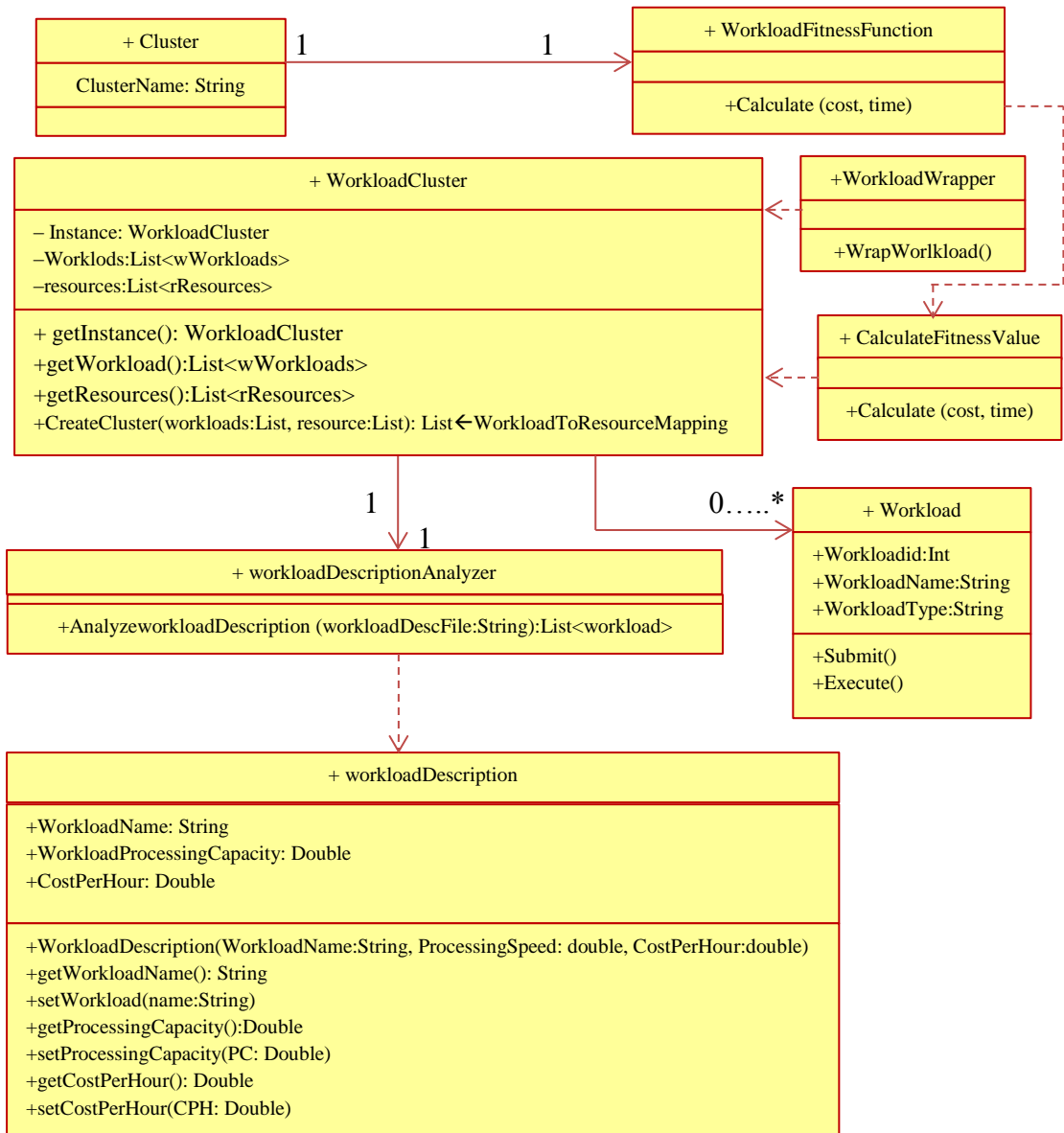


Figure 4.24: Class Diagram for Workload Allocation

4.8 Conclusion

This chapter discussed the design of proposed solution through a Cloud Workload Management Framework. Workload analysis and identification has been described. The classification of these Cloud workloads is done through K-Means Clustering Algorithm by assigning the appropriate weights to the different quality attributes. The design pattern template used to collect the information related to specific problem has been developed and discussed. The design patterns has been identified and matched with cloud workloads. In the last section, the execution design of Cloud workloads has been presented. Next chapter discusses the Proposed Policies for Cloud Workloads.

Proposed Policies for Cloud Workloads

Cloud workload have been identified and classified in the last chapter. On the basis of resource requirement pattern the clustered workloads need to be scheduled. This chapter presents the proposed workload scheduling policies. Decision tree is used to select the appropriate policy based on workload details described by Cloud consumer. The four resource scheduling policies are Compromised Cost - Time Based (CCTB) Scheduling Policy, Time Based (TB) Scheduling Policy, Cost Based (CB) Scheduling Policy and Bargaining Based (BB) Scheduling Policy.

5.1 Resource Scheduling Procedure

Four resource scheduling policies (Compromised Cost - Time Based (CCTB) Scheduling Policy, Time Based (TB) Scheduling Policy, Cost Based (CB) Scheduling Policy and Bargaining Based (BB) Scheduling Policy) are proposed in this thesis. Decision tree is used to select the appropriate policy based on workload details described by Cloud consumer (Discussed in previous chapter). Cloud environment and a scheduler that implements different scheduling policies based on the decision taken by Cloud provider. Resource Scheduling Procedure is shown in Figure 5.1.

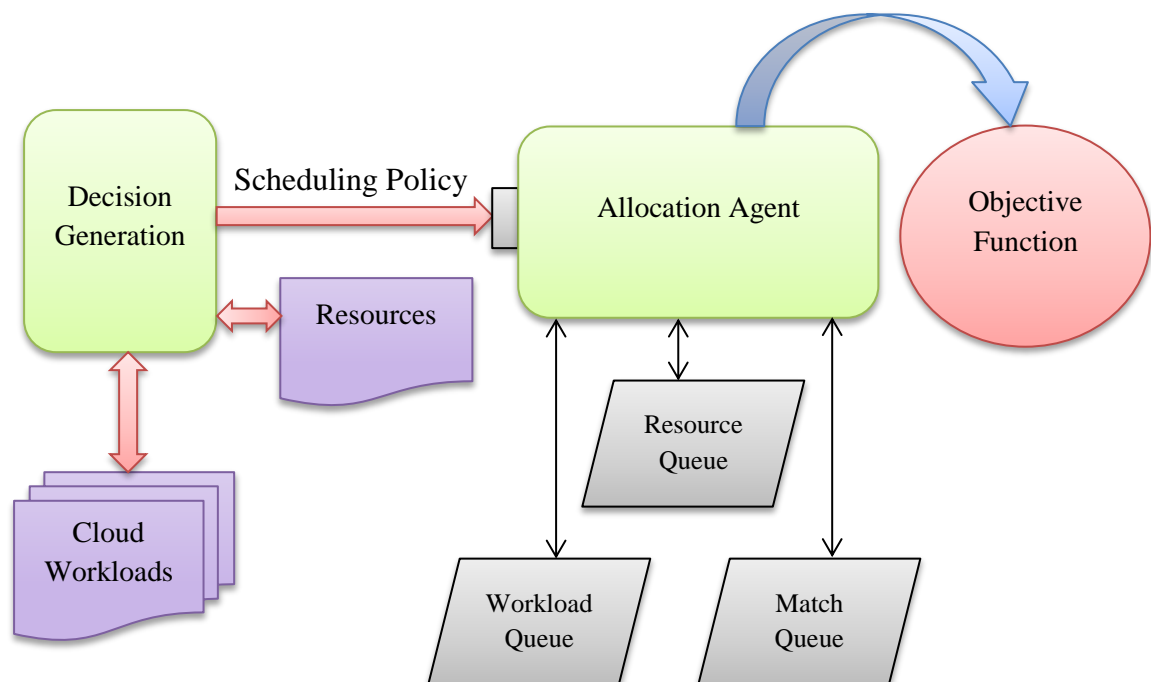


Figure 5.1: Resource Scheduling Procedure

Based on the scheduling policy, the resources are allocated to the Cloud workloads. The information of the Cloud workloads and computational resources is sent to the allocation agent. The allocation agent implements four resource scheduling policies: Compromised Cost - Time Based (CCTB) Scheduling Policy, Time Based (TB) Scheduling Policy, Cost Based (CB) Scheduling Policy and Bargaining Based (BB) Scheduling Policy. Cloud Workload Management Portal (CWMP) produces the Cloud workloads. It calculates workload deadline time. Each workload is characterized by their deadline, estimated budget and policy. The QoS of each Cloud workload is also represented in the scheduling request of the Cloud workload. Similarly, QoS, such as processing speed, is generated for each computational resource.

5.1.1 Cloud Workload Attributes

3000 independent Cloud workloads were generated randomly in CloudSim as Cloudlets. For each Cloud workload, the attributes of the Cloud workload include deadline, estimated budget and scheduling policy. The resource scheduling policies consider only one dimensional QoS (processing speed). The QoS is generated to be 1 to 32 with the ratio following given distribution of the QoS request. In this thesis, the six scenarios of the QoS distribution has been defined (See Section 6.5).

5.1.2 Resource Attributes

For each set of resources, the attributes of the resource include number of resources (or computing nodes), QoS provided, and the information of each resources, which includes the deadline, estimated budget and scheduling policy. All configurations about the resources will remain the same during the experiment. In this thesis, one-dimensional QoS (Processing Speed) has been implemented.

5.1.3 Allocation Agent

The allocation agent receives the Cloud workloads and puts them into the workload queue. While the workload queue is not empty, the allocation starts the scheduling policy to find the right workload-resource match according to the policy. To compare proposed QoS guided policies with the existing scheduling policies, all the proposed policy has been implemented to get the performance data.

5.2 Decision Tree Based Scheduling Criteria

Classification Tree analysis is when the predicted outcome is the class to which the data belongs. Regression Tree analysis is when the predicted outcome can be considered a real number.

5.2.1 Problem Statement

The classification and regression objective [142] requires a set

$$I = \{I_1, \dots \dots I_n\}$$

where $I_j, 1 \leq j \leq n, j \in \mathbb{N}$ is an object being reviewed and n is a rational number. The objects may be the information on executing policies in different values of cost and time (See Table 5.1). Each object is characterized with a set of variables:

$$I_i = \{x_1, \dots \dots x_m, y\}$$

Where the x_j are independent attributes, for which the values are known, and on which the value of the target variable y is based and m is total number of independent attributes. The independent attributes are: Cost and Time. The target variable is Policy. A set of independent attributes is often defined as a vector:

$$X = \{x_1, \dots \dots x_m\}$$

Every single variable x_j can acquire values from a certain set:

$$C_j = \{c_{j1}, c_{j2} \dots \dots \dots\}$$

If the values of a variable are elements of a finite set, it is denoted as a categorical variable. For example, the variable Policy acquires values from range {Compromised Cost - Time Based, Time Based, Cost Based, Bargaining Based}.

If a range $C_y = \{c_1, c_2 \dots \dots \dots\}$ of the variable y is finite, then the objective is referred to as classification objective, else the objective is referred to as regression objective. The policy details are shown in Table 5.1.

Table 5.1: Policies Description

Cost	Time	Policy
Minimum	Minimum	Compromised Cost - Time Based
Maximum	Minimum	Time Based
Minimum	Maximum	Cost Based
Cost Agreement	Time Agreement	Bargaining Based

5.2.2 Decision Trees

Decision trees comprise a way of presenting rules in a hierarchical, sequential structure. Figure 5.2 shows a decision tree for the data presented in Table 5.1.

5.2.3 Mathematical Functions

In this case, the objects being reviewed are considered as points in an $(m + 1)$ – dimensional space. Then, the variables of an object $I_j = \{x_1, \dots \dots x_m, y\}$, are considered as coordinates, which are set into relation in the following function:

$$y = \{w_0 + w_1x_1 + \dots + w_mx_m\},$$

where w_0, \dots, w_m — weights of independent attributes, where the determination of these weights is the central objective of finding a classification function.

It is obvious that all variables should be presented as numeric parameters. There are different ways to transform logical and categorical variables into numeric ones. Logical types, as a rule, are encoded by the figures 1 and 0.

The values of categorical variables are the names of the possible categories of an object being reviewed. Indeed, there may be more than two of such states. Their names should be mentioned and enumerated in a list. Each name from the list may be presented by its individual number. For the transformation into a numeric variable, for example, the value of the variable Policy = {Compromised Cost - Time Based, Time Based, Cost Based, Bargaining Based} may be replaced by the values {0, 1, 2, 3}. It is observed from the results obtained in experimentation that the Naive Bayes model is quite appealing [97] because of its simplicity, elegance, robustness and effectiveness (See Section 2.4).

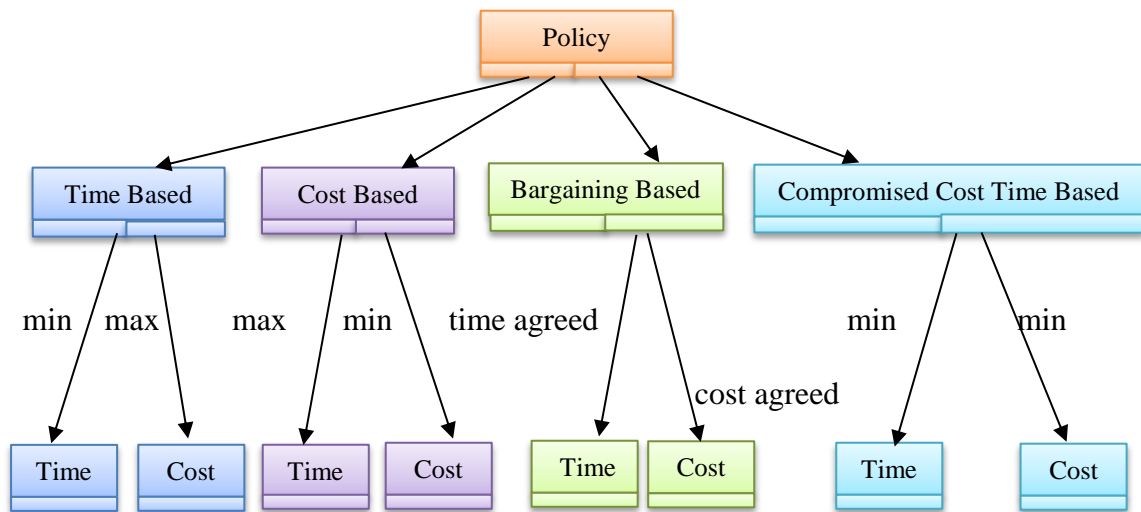


Figure 5.2: Decision Tree of Scheduling Policies

Such a classification can be made, via a Naive Bayes algorithm [96], which applies the Bayes' formula [98] for the calculation of a probability. The term Naive is derived from the Naive assumption that all attributes reviewed are independent from each other [96]. Let us denote the probability that a certain object I_j belongs to a class C_j (that is $y = c_j$) as $P(y = c_j)$. An event corresponding to the equality of independent attributes to specific values is denoted as E , with the probability of its occurrence as $P(E)$. The idea of the algorithm is calculating the conditional probability of affiliation

of an object to c_j , with its independent attributes being equal to specific values. With the use of Bayes' theorem:

$$P(y = c_j | E) = P(E | y = c_j) \cdot P(y = c_j) / P(E)$$

In other words, rules are built with a conditional part that compares all independent attributes with the corresponding possible values. In the concluding part all possible values of the target variable are present:

$$\text{If } [(x_1 = c_{j_1}) \text{ and } \dots \text{ and } (x_m = c_{j_m})] \text{ then } (y = c_j)$$

For each of these rules, Bayes' formula calculates a corresponding probability. Assuming that attributes acquire values independently from each other, let us perform the calculation of the probability $P(E | y = c_j)$ through a multiplication of probabilities for each independent attribute:

$$P(E | y = c_j) = P(x_1 = c_{j_1} | y = c_j) \dots P(x_m = c_{j_m} | y = c_j)$$

Then, the probability for the whole rule can be defined by formula:

$$P(y = c_j / E) = P(x_1 = c_{j_1} / y = c_j) \times \dots \times P(x_m = c_{j_m} | y = c_j) \times P(y = c_j) / P(E)$$

The probability of affiliation of an object to class c_j , provided that its attribute x_j is equal to c_{j_i} is defined by the formula:

$$P(x_j = c_{j_i} / y = c_j) = P(x_j = c_{j_i} \cap y = c_j) / P(y = c_j)$$

which is the ratio of the number of objects in the training sample, in which $x_j = c_{j_i}$ and $y = c_j$, and the number of objects belonging to class C_j . For example, for the objects from Table 5.1, the following probabilities for the values of the independent attribute Cost and Time has been obtained:

$$P(\text{Cost} = \text{Minimum} | \text{Policy} = \text{Compromised Cost - Time Based}) = 1/4;$$

$$P(\text{Cost} = \text{Maximum} | \text{Policy} = \text{Time Based}) = 1/4;$$

$$P(\text{Cost} = \text{Minimum} | \text{Policy} = \text{Cost Based}) = 1/4;$$

$$P(\text{Cost} = \text{Cost Agreed} | \text{Policy} = \text{Bargaining Based}) = 1/4;$$

$$P(\text{Time} = \text{Minimum} | \text{Policy} = \text{Compromised Cost - Time Based}) = 1/4;$$

$$P(\text{Time} = \text{Minimum} | \text{Policy} = \text{Time Based}) = 1/4;$$

$$P(\text{Time} = \text{Maximum} | \text{Policy} = \text{Cost Based}) = 1/4;$$

$$P(\text{Time} = \text{Time Agreed} | \text{Policy} = \text{Bargaining Based}) = 1/4;$$

Thus, if it is necessary to define whether a policy would take place with the following values of independent attributes (Event E):

Cost = Maximum;

Time = Maximum;

One should calculate the following conditional probabilities:

$$P(\text{Policy} = \text{Compromised Cost - Time Based} \mid E) = P(\text{Cost} = \text{Maximum} \mid \text{Policy} = \text{Compromised Cost - Time Based}) \times P(\text{Time} = \text{Maximum} \mid \text{Policy} = \text{Compromised Cost - Time Based}) / P(E);$$

$$P(\text{Policy} = \text{Time Based} \mid E) = P(\text{Cost} = \text{Minimum} \mid \text{Policy} = \text{Time Based}) \times P(\text{Time} = \text{Maximum} \mid \text{Policy} = \text{Time Based}) / P(E);$$

$$P(\text{Policy} = \text{Cost Based} \mid E) = P(\text{Cost} = \text{Maximum} \mid \text{Policy} = \text{Cost Based}) \times P(\text{Time} = \text{Minimum} \mid \text{Policy} = \text{Cost Based}) / P(E);$$

$$P(\text{Policy} = \text{Bargaining Based} \mid E) = P(\text{Cost} = \text{Maximum} \mid \text{Policy} = \text{Bargaining Based}) \times P(\text{Time} = \text{Maximum} \mid \text{Policy} = \text{Bargaining Based}) / P(E);$$

By inserting the corresponding probabilities, the following figure has been obtained:

$$P(\text{Policy} = \text{Compromised Cost - Time Based} \mid E) = 1/4 \times 1/4 / P(E) = 0.0625/P(E);$$

$$P(\text{Policy} = \text{Time Based} \mid E) = 1/4 \times 1/4 / P(E) = 0.0625/P(E);$$

$$P(\text{Policy} = \text{Cost Based} \mid E) = 1/4 \times 1/4 / P(E) = 0.0625/P(E);$$

$$P(\text{Policy} = \text{Bargaining Based} \mid E) = 1/4 \times 1/4 / P(E) = 0.0625/P(E);$$

The description of decisions for selection of scheduling policy is shown in Figure 5.3.

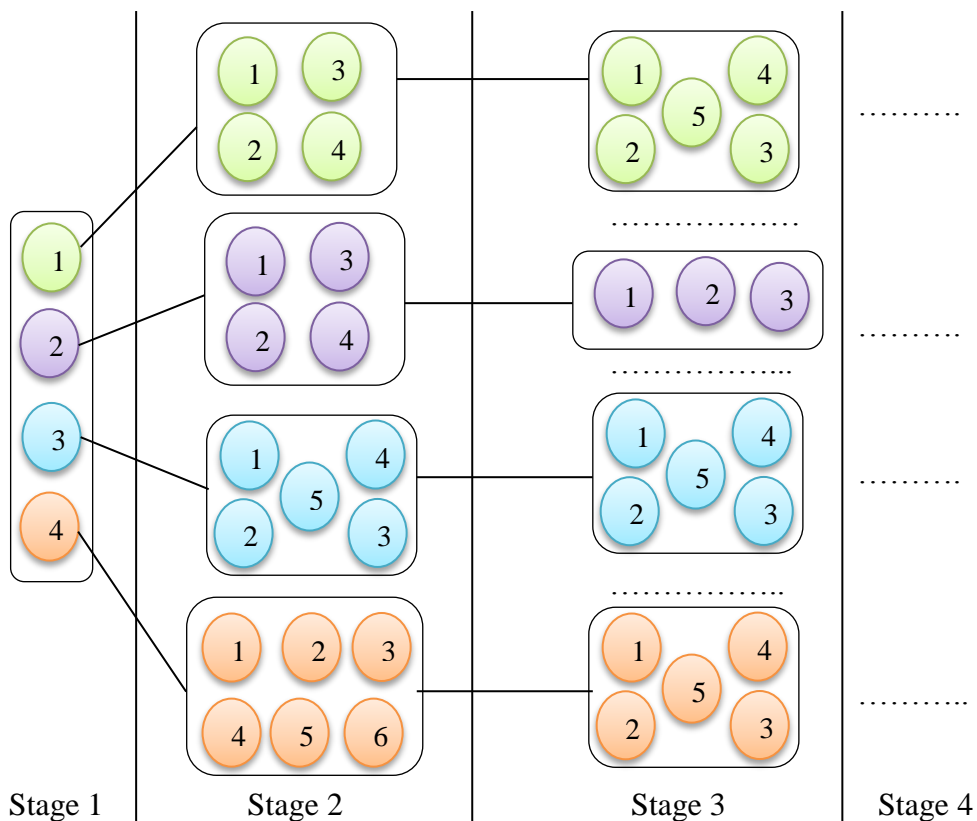


Figure 5.3: Decision Tree Based Branching

The probability P(E) is not accounted for, as in the normalization of the probabilities for each of the possible rules, it vanishes. The normalized probability for a rule is calculated by formula:

$$P'(y=c_j | E) = P(y=c_j | E) / \sum P(y=c_j | E)$$

In this case, one may say that under specified conditions, the Compromised Cost - Time Based, Time Based, Cost Based, Bargaining Based Policy will select with a probability of:

$$\begin{aligned} P(\text{Policy} = \text{Compromised Cost - Time Based} | E) &= \\ 1/4 \times 1/4 / (0.0625+0.0625+0.0625+0.0625) &= 0.0625/ 0.25 = 0.25; \\ P(\text{Policy} = \text{Time Based} | E) &= 1/4 \times 1/4 / P(E) = 0.0625/ 0.25 = 0.25; \\ P(\text{Policy} = \text{Cost Based} | E) &= 1/4 \times 1/4 / P(E) = 0.0625/ 0.25 = 0.25; \\ P(\text{Policy} = \text{Bargaining Based} | E) &= 1/4 \times 1/4 / P(E) = 0.0625/ 0.25 = 0.25; \end{aligned}$$

The rules for four resource scheduling policies along with conditions are discussed below:

5.2.4 Compromised Cost - Time Based Scheduling Policy

If [(Workload Pending = Yes) and (TECT \leq D_t = True) and (TEC \leq B_E = True) and (E_{cloud} \leq E_{Threshold} = True)] then (Policy = Yes)

If [(Workload Pending = Yes) and (TECT \leq D_t = True) and (TEC \leq B_E = False) and (E_{cloud} \leq E_{Threshold} = True)] then (Policy = No)

If [(Workload Pending = Yes) and (TECT \leq D_t = False) and (TEC \leq B_E) = True) and (E_{cloud} \leq E_{Threshold} = True)] then (Policy = No)

If [(Workload Pending = Yes) and (TECT \leq D_t = False) and (TEC \leq B_E = False) and (E_{cloud} \leq E_{Threshold} = True)] then (Policy = No)

If [(Workload Pending = No) and (TECT \leq D_t = (True \vee False) and (TEC \leq B_E = (True \vee False)) and (E_{cloud} \leq E_{Threshold} = True \vee False))] then (Policy = No)

The above discussed rules are described in Table 5.2 along with conditions.

Table 5.2: Compromised Cost - Time Based Rules

Pending Workload	TECT \leq D _t	TEC \leq B _E	E _{cloud} \leq E _{Threshold}	Policy
Yes	True	True	True	Yes
Yes	True	False	True	No
Yes	False	True	True	No
Yes	False	False	True	No
No	-	-	-	No

Where TECT = Total Expected Completion Time, D_t = Deadline Time, TEC = Total Expected Cost, B_E = Estimated Budget, E_{Cloud} = Actual Energy Consumption and $E_{Threshold}$ = Threshold Energy Value.

5.2.5 Time Based Scheduling Policy

If [(Workload Pending = Yes) and (Urgency = Yes) and (Add Resource = Reserve)] then (Workload = Admit)

If [(Workload Pending = Yes) and (Urgency = No) and (Add Resource = Available)] then (Workload = Admit)

If [(Workload Pending = No) and (Urgency = (True v False)) and (Add Resource = (True v False))] then (Workload = Admit)

The above discussed rules are described in Table 5.3 along with conditions.

Table 5.3: Time Based Rules

Workload Pending	Urgency	Add Resource	Workload
Yes	Yes	Reserve	Admit
Yes	No	Available	Admit
No	-	-	Finish

5.2.6 Cost Based Scheduling Policy

If [(Workload Pending = Yes) and ($R_A > 0$ = True) and ($E_t > W_d$ = True) and ($B_A > P_r$ = True)] then (Status = Admit Workload)

If [(Workload Pending = Yes) and ($R_A > 0$ = False) and ($E_t > W_d$ = True) and ($B_A > P_r$ = True)] then (Status = Add Resource)

If [(Workload Pending = No) and ($R_A > 0$ = (True v False)) and ($E_t > W_d$ = (True v False)) and ($B_A > P_r$ = (True v False))] then (Status = Finish)

If [(Workload Pending = Yes) and ($R_A > 0$ = True) and ($E_t > W_d$ = False) and ($B_A > P_r$ = True)] then (Status = Finish)

If [(Workload Pending = Yes) and ($R_A > 0$ = True) and ($E_t > W_d$ = True) and ($B_A > P_r$ = False)] then (Status = Finish)

The above discussed rules are described in Table 5.4 along with conditions.

Table 5.4: Cost Based Rules

Workload Pending	$R_A > 0$	$E_t > W_d$	$B_A > P_r$	Status
Yes	True	True	True	Admit Workload
Yes	False	True	True	Add Resource
No	-	-	-	Finish
Yes	True	False	True	Finish
Yes	True	True	False	Finish

Where R_A = Resource Available, E_t = Estimated Time, P_r = Resource Price, W_d = Desired Deadline and B_A = Available Budget.

5.2.7 Bargaining Based Scheduling Policy

If [(Workload Pending = Yes) and (Slot Available = Yes) and ($Cur_t \leq NST = True$) and ($t.value() \neq g.value() = True$) and ($TECT \leq D_t = True$)] then (Mapping = Yes)

If [(Workload Pending = No) and (Slot Available = Yes) and ($Cur_t \leq NST = True$) and ($t.value() \neq g.value() = True$) and ($TECT \leq D_t = True$)] then (Mapping = No)

If [(Workload Pending = Yes) and (Slot Available = No) and ($Cur_t \leq NST = True$) and ($t.value() \neq g.value() = True$) and ($TECT \leq D_t = True$)] then (Mapping = No)

If [(Workload Pending = Yes) and (Slot Available = Yes) and ($Cur_t \leq NST = False$) and ($t.value() \neq g.value() = False$) and ($TECT \leq D_t = True$)] then (Mapping = Yes)

If [(Workload Pending = Yes) and (Slot Available = Yes) and ($Cur_t \leq NST = True$) and ($t.value() \neq g.value() = True$) and ($TECT \leq D_t = False$)] then (Mapping = No)

The above discussed rules are described in Table 5.5 along with conditions.

Table 5.5: Bargaining Based Rules

Workload Pending	Slot Available	$Cur_t \leq NST$	$t.value() \neq g.value()$	$TECT \leq D_t$	Mapping
Yes	Yes	True	True	True	Yes
No	Yes	True	True	True	No
Yes	No	True	True	True	No
Yes	Yes	False	False	True	No
Yes	Yes	True	True	False	No

Where Cur_t = Current Time, NST = Next Scheduled Time, $t.value()$ = Workload Price Taken, $g.value()$ = Workload Price Given, $TECT$ = Total Expected Completion Time and D_t = Deadline Time.

5.3 Resource Scheduling Policies

5.3.1 FCFS Based Scheduling Policy

FCFS Based Scheduling Policy allocates the resources to the Cloud workloads without considering the minimum processing time and execution cost. This algorithm is already implemented in CloudSim in VMScheduler Class.

5.3.2 Compromised Cost - Time Based (CCTB) Scheduling Policy

In this scheduling policy, Cloud provider minimizing cost as well as execution time along with least energy consumption. It calculates the TEC, TECT and Time

Difference (T_d) to allocate the resources. The allocation agent find the missed deadlines and calculate Time Difference for each workload then use the extra available time to the workloads with missed deadlines and execute all the Cloud workloads within their corresponding deadlines. The Compromised Cost - Time Based (CCTB) Scheduling Policy is shown in Figure 5.4.

Objective Function – The goal of an objective function that goals to decrease the sum of product of cost and time expended for finishing all n workloads of a given Bulk of Workloads (BoW). This objective function (min z) successfully captures the compromise between performance improvement and additional price related with public resource payment. Further formally, the workload assignment problem with the cost function of each resource can be generally formulated as follows:

$$\min z = \sum_{m=1}^n (E_t)_m \times (B_H)_m$$

Where, m is a current workload that is being executed, E_t = Estimated Time and B_H = Budgeted per Hour. The goal of Cloud provider is to maximize the resource utilization and minimize the actual energy consumption. The Cloud workload will be executed only when the Actual Energy Consumption is less than the Threshold Energy Value.

The energy model is devised on the basis that processor utilization has a linear relationship with energy ingestion. For a particular Cloud workload, the information on its processing time and processor utilization is sufficient to measure the energy consumption for that Cloud workload. For a resource r_a at any given time, the utilization U_a is defined as

$$U_a = \sum_{b=1}^c U_{a,b}$$

where c is the number of Cloud workloads running at that time and $U_{a,b}$ is the resource usage of a Cloud workload w_a . The actual energy consumption E_{cloud} of a resource r_a at any given time is defined as

$$E_{cloud} = (PC_{max} - PC_{min}) \times U_a + PC_{min}$$

where PC_{max} is the power consumption at the peak load (or 100% utilization) and PC_{min} is the minimum power consumption in the active mode (or as low as 1% utilization).

Total Expected Completion Time (TECT)	$TECT = C_t + P_t$
Deadline Time (D_t)	$D_t = W_d - Cur_t$
Total Expected Cost (TEC)	$TEC = C_c + C_{min}$

The complexity of CCTB Scheduling Policy is influenced by number of change points (C) i.e. rescheduling and requested resources (r) of *Workload* being scheduled. Here,

$$C = (S_t + T_e + T_s + R_t)$$

where,

S_t = start times of all workloads	T_s = suspend times of all workloads
T_e = end times of all workloads	R_t = resume times of all workloads

Consider lesser pre-emption as its objective. The complexity of the algorithm mainly depends on two important objectives:

- Minimize the rejection rate of the incoming requests.
- Minimize reshuffle cost (avoid rescheduling of already accommodated leases as much as possible).

Policy: Compromised Cost-Time Based (CCTB) Scheduling Policy
<p>Data: Name of Workload Type of workload Desired Deadline (W_d) Estimated Budget (B_E) Preferred Policy</p> <p>Result: Each workload will be mapped to the resources within available budget and desired deadline as per the specified policy.</p>

Begin:

Intilize resourceList [No. of Resources]

Intialize workloadList [No. of Workloads]

resourceList = getAvailbleResource()

workloadList = getWorkloadtoSchedule()

Step 1:

a) Group Workloads into two categories:

- Homogenous Workload
- Heterogeneous Workload

Homogenous workloads based on Key QoS as well as some relationship among them (Dependency of workload (s) to another workload (s))

b) Evaluate the minimum processing time and price for every workload from the available set of resources by using cost and time fitness formulas.

c) Calculate the Total Expected Completion Time (TECT)

d) Calculate Deadline Time (D_t)

e) Calculate the Total Expected Cost (TEC)

f) If ($TECT \leq D_t$ && $TEC \leq B_E$)

{

 If ($E_{cloud} \leq E_{Threshold}$)

 {

 Dispatch the workload

 }

 else

 {

 Redistribute workloads

 }

 }

else

{

 Change the Scheduling Policy

}

Step 2:

a) Allot the Cloud customer whole deadline and budget into every workload

partition in proportion to their least processing time and cost respectively calculated in step 1 (b).

The deadline and budget is spread according to following rule: Processing time of workload may differ. Some workload may require only small time to be completed and some need at least more than 2 hours.

There are many probable execution time and cost for each workload but use only P_t and C_{min} to allocate the overall deadline and given budget correspondingly.

- b) Sorting the entire resource list by giving highest priority to the expensive one based on their cost.

Step 3:

Select a resource to process a specific workload from the workload list so that processing time and expenses for a specific Cloud workload must be less than desired deadline W_d and estimated budget value (B_E) correspondingly.

Step 4:

Repeat the step 3 until all the Cloud workloads within both the partition have been scheduled and executed, otherwise rescheduling the Cloud workloads to the resources.

Figure 5.4: Compromised Cost-Time Based (CCTB) Scheduling Policy

The abbreviations used in the Compromised Cost-Time Based (CCTB) Scheduling Policy are given below:

E_{cloud}	= Actual Energy Consumption
$E_{Threshold}$	= Threshold Energy Value
C_t	= Communication Time
P_t	= Minimum Processing Time
W_d	= Desired Deadline
Cur_t	= Current Time
C_c	= Communication Cost
C_{min}	= Minimum cost of a particular workload
B_E	= Estimated Budge

5.3.3 Cost Based (CB) Scheduling Policy

First, the allocation agent begins to compute the cost of each Cloud workload then sort, as the priority is given to the Cloud workload which has maximum budget. If the two workloads have same budget then that workload will execute first that has lesser execution time. By default, PS=1. The allocation agent then schedules all the

workloads with high budget request to the resources that provide high QoS. Finally, all other workloads are scheduled on the available resources set. The Cost Based (CB) Scheduling Policy is shown in Figure 5.5.

The fitness value (Estimated Cost) is calculated as follows:

$$\text{Estimated Cost: Budgeted/Hour } (B_H) \quad B_H = \frac{B_E}{W_d - Cur_t}$$

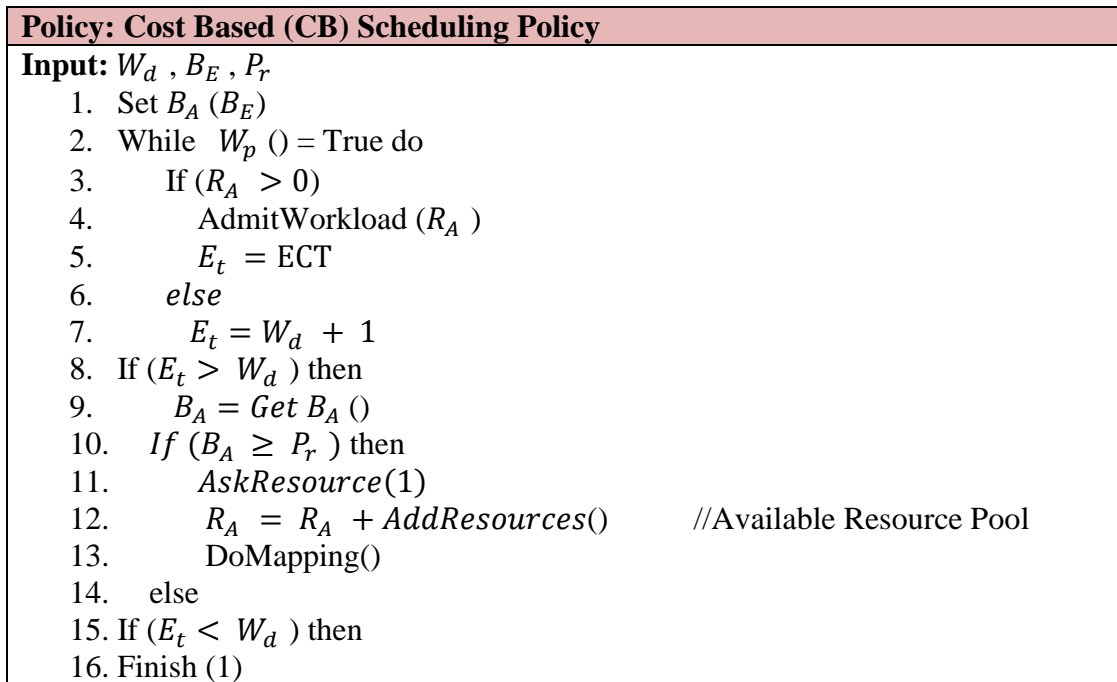


Figure 5.5: Cost Based (CB) Scheduling Policy

The abbreviations used in the Cost Based (CB) Scheduling Policy are given below:

B_H	= Budget per hour
W_d	= Desired Deadline
B_E	= Estimated Budget
P_r	= Resource Price
Cur_t	= Current Time
R_A	= Resource Available
W_p	= Workload Pending
B_A	= Available Budget
ECT	= Estimated Completion Time
E_t	= Estimated Time

To maximize the chance that the desired deadline can still be met after terminating one resource, termination is only done if the estimated completion time is lesser than a desired deadline ($E_t < W_d$). In the current implementation of Cost Based Scheduling Policy, consider as a constant coefficient.

5.3.4 Time Based Scheduling Policy

First, the allocation agent begins to compute the Deadline Time of the Cloud workload in the given budget. Allocate Resources based on time, the workload which has shortest Deadline Time (D_t) will execute first. If the two workloads have same deadline time then that workload will execute first that has lesser execution time. By default, PS=1. The allocation agent then schedules all the Cloud workloads with smallest execution time request to the resources that provide high QoS. If any deadline found missed then recalculate the execution time by increasing the value of Processing Speed (PS) and it will increase cost only. The Time Based (TB) Scheduling Policy is shown in Figure 5.6. The fitness value (Estimated Time) is calculated as follows:

For Homogenous Workloads
Time estimation(E_t) = Workload Remained * Workload Runtime
For Heterogeneous Workloads
Time estimation (E_t) = $\sum_{i=1}^n W_i \text{Runtime}$

Policy: Time Based (TB) Scheduling Policy
<p>Input: W_d, B_E, P_r</p> <ol style="list-style-type: none"> 1. Calculate B_H $B_H = \frac{B_E}{W_d - Cur_t}$ 2. $askCount = \frac{B_H}{P_r}$ 3. askResource (askCount) 4. if (Criteria == Urgency) <ul style="list-style-type: none"> { <li style="padding-left: 20px;">$R_A = R_A + AddReserveResources()$ //Reserve Resource Pool } else { <li style="padding-left: 20px;">$R_A = R_A + AddResources()$ //Available Resource Pool } 5. DoMapping based on Criteria 6. While $W_p() = True$ do 7. AdmitWorkload (R_A) 8. Finish (askCount)

Figure 5.6: Time Based (TB) Scheduling Policy

The abbreviations used in the Time Based (TB) Scheduling Policy are given below:

B_H	= Budget per hour
W_d	= Desired Deadline
B_E	= Estimated Budget
P_r	= Resource Price
Cur_t	= Current Time
R_A	= Resource Available
W_p	= Workload Pending

DoMapping(), in both Cost and Time Scheduling Policies, takes care of budgeting for hired resources and decreases the available budget base on the price of the hired resources per hour. If there is not enough budget, then DoMapping() terminates each hired resource before it starts a new mapping cycle [143].

5.3.5 Bargaining Based (BB) Scheduling Policy

The implementation complies with the negotiation among the various resources and Cloud workload producer along with different time slots. The allocation agent allocates the resources based on the bargaining between them. The Bargaining Based (BB) Scheduling Policy is shown in Figure 5.7.

The different formulas were derived for resource allocation in Grid [143] [144] [145] [146]. After modification in these formulas and made it suitable for resource allocation in Cloud.

- Deadline Urgency - Deadline urgency (D_u), which specifies Cloud customer urgency to get workload (s) completed, is defined as:

$$D_u = \frac{[W_d - S_t]}{[P_t]} - 1$$

Where S_t is the start time of the user application, W_d is Desired Deadline and P_t is the execution time of Cloud customer's workload. The deadline is considered very urgent when $D_u < 0.25$, intermediate when $0.25 < D_u < 0.75$ and relaxed when $D_u > 0.75$. This metric shows how the scheduler deals with Cloud customer with different requirement on time.

- Budget per Workload - The budget (valuation) provided by the Cloud customer for their workload is divided by the number of workloads contained within the application to normalize the budget across all the workloads. This metric examines how the schedulers allocate resources fairly among different Cloud customer with different budget groups.

Amount of Deadlines Lost with Rise in Quantity of Cloud Customer Workloads - This metric is used to examine how the scheduling algorithms are able to cope up with multiple Cloud customers when requirement for resources exceeds their availability. To evaluate the benefit of resource scheduling mechanism for the resources the following metrics has been considered:

- Average Load of a Resource (R_{AL}) - The number of resources (R_n) used through all the queues offered for the resource scheduler, divided by the total number of resources (R_a) allocated to these queues.

$$R_{AL} = \frac{R_n}{R_a}$$

- Average Estimation of a Resource (E_{AV}) - The average of the estimations (E_{AV}) assigned to the queues in a resource by the resource scheduler.

$$E_{AV} = \frac{E_1, E_2, \dots \dots \dots E_n}{n}$$

- Estimation (Set a price) of Resources - In order to balance load through independent Cloud services, the resource scheduler tries to admit more workloads to the least allocated resources. The supreme urgent workload must be coordinated to the fastest queue. The estimation of resources must be such that the resource with min load would get min value (the max given is matched to minimum taken) [99]. Therefore, E_{cr} (time), the Estimated cost of a resource, is identified as follows:

E_{cr} (time) \propto $Wait_r$ (time-1) , where $Wait_r$ (time-1) is average queue waiting time,

$$E_{cr}$$
 (time) \propto $\frac{[Requirement]}{[Availability]}$,

E_{cr} (time) \propto I_r , where I_r is initial price given by the resource,

E_{cr} (time) \propto $Load$ (time-1) , where is $Load$ (time-1) load of resource,

After combining above equations, the estimated cost metric for the clusters are:

$$E_{cr}(\text{time}) = c \times Wait_r(\text{time}-1) \times I_r \times Load_{(\text{time}-1)}$$

where c is a proportionality constant.

- Estimation (Set a price) of Cloud Customer Workload - As discussed previously, each Cloud customer admits to the resource scheduler estimated budget (B_E), desired deadline (W_d), workload size (S_w), and the number of resources required (R_u). Let $E_u(t)$ be the estimation of the Cloud customer workload. As Cloud customer workload with the maximum budget must be given higher priority,

$$E_u(t) \propto B_E$$

Also, the more urgent a workload, higher its priority, therefore,

$$E_u(t) \propto \frac{1}{[W_d - Cur_t]}$$

where Cur_t is the current time. A Cloud customer workload that has waited longer gains higher priority.

$E_u(t) \propto W_d - Admit_{time}$ i, where $Admit_{time}$ is the time the workload was admitted.

Finally, as the estimation is based on the requirement and availability of resources (clusters) in the system,

$E_u(t) \propto \frac{[Requirement]}{[Availability]}$, where requirement is total number of workload to allocate and availability is the total number of CPUs in all resources.

Therefore, the following metric for estimation of Cloud consumer workload has been obtained,

$$E_u(t) = C_u \times B_E \times \frac{1}{[W_d - Cur_t]} \times \frac{[Requirement]}{[Availability]} \times W_d - Admit_{time}$$

where C_u is a proportionality constant.

Policy: Bargaining Based (BB) Scheduling Policy

```
1. MappingList  $\leftarrow$  empty
2. While  $Cur_t \leq NST$  do
3.     IncomingResource ( $A_y$ ) // From Cloud Providers
4.     IncomingWorkload ( $W_y$ ) // From Cloud Customers
5. end while
6. CalculateAvailable( $W_y$ )
7. CalculateRequirement( $A_y$ )
8. UpdateGivenCost ( $W_y$ )
9. UpdateTakenCost ( $A_y$ )
10. ListTaken  $\leftarrow$  Sort_Taken( $A_y$ )
11. ListGiven  $\leftarrow$  Sort_Given( $A_y$ )
12. Let  $y=0$  and  $s=$  availableQueueSlots( $t$ )
13. for every Given  $x$  in the list Given do
14.      $g \leftarrow$  given( $x$ ) for each workload  $y$  of Given  $g$  do
15.         if( $s \neq 0$ ) then
16.             If  $t.value() \neq g.value()$  then
17.                 If Check_Desired_Deadline ( $x, t$ )
18.                     mapping  $y =$  AllocateResource ( $x, t$ )
19.                     addingMappingList (Map_List)
20.                      $s--$ 
21.                 else
22.                     Goto line number 36
23.                 endif
24.             else
25.                 If IsEmpty (workloads) then
26.                     break
27.                 else
28.                      $y++$ 
29.                      $t \leftarrow$  taken( $y$ )
30.                 endif
31.             endif
32.         else
33.         endif
34.     end for every
35. end for every
36. for each instance in Map_List do
37.     IntimateCloudCustomer()
38. end for every
```

Figure 5.7: Bargaining Based (BB) Scheduling Policy

The abbreviations used in the Bargaining Based (BB) Scheduling Policy are given below:

Cur_t = Current Time NST = Next Schedule Time
--

5.4 Conclusion

This chapter discussed the proposed workload scheduling policies. Decision tree has been used to select the appropriate policy based on workload details described by Cloud consumer. The four resource scheduling policies (Compromised Cost - Time Based (CCTB) Scheduling Policy, Time Based (TB) Scheduling Policy, Cost Based (CB) Scheduling Policy and Bargaining Based (BB) Scheduling Policy) have been proposed and discussed in this chapter. Next chapter discusses the Implementation and Experimental Results.

Implementation and Experimental Results

This chapter focuses on the implementation of the proposed framework followed by framework execution. CloudSim 3.0 is used to validate the results of the proposed scheduling policies. The implementation of Cloud Workload Management Portal (CWMP) describes in this chapter. WEKA tool is used to present the results of clustering of Cloud workloads. To evaluate the performance, proposed scheduling policies compares with existing policies.

6.1 Tools for Setting Cloud Environment

A brief introduction of tools which are used in designing and implementation of proposed framework is given below.

6.1.1 Microsoft Visual Studio 2010

Microsoft Visual Studio 2010 is an Integrated Development Environment (IDE) from Microsoft [147]. It is used to develop console and graphical user interface applications along with Windows Forms, web sites, web applications, and web services in both native code together with managed code for all platforms supported by Microsoft Windows. Visual Studio 2010 comes with .NET Framework 4 and supports developing applications targeting Windows 7.

6.1.2 NetBeans IDE 7.1.2

The NetBeans IDE 7.1.2 is a modular, standards-based, Integrated Development Environment (IDE) written in the Java programming language [148]. The NetBeans IDE 7.1.2 project consists of an open source IDE and an application platform, which can be used as a generic framework to build any kind of application. The focus of NetBeans IDE 7.1.2 is improved developer productivity through a smarter, faster editor, and the integration of all NetBeans products into one IDE. NetBeans IDE 7.1.2 is an Integrated Development Environment (IDE) for developing primarily with Java.

6.1.3 CloudSim 3.0

The CloudSim toolkit supports both system and behaviour modelling of Cloud system components such as data centres, Virtual Machines (VMs) and resource provisioning policies. It implements generic application provisioning techniques that can be

extended with ease and limited effort. Currently, it supports modelling and simulation of Cloud Computing environments consisting of both single and inter-networked Clouds (federation of Clouds). Moreover, it exposes custom interfaces for implementing policies and provisioning techniques for allocation of VMs under inter-networked Cloud Computing scenarios [149].

The main advantages of using CloudSim for initial performance testing include: (i) time effectiveness: it requires very less effort and time to implement Cloud-based application provisioning test environment and (ii) flexibility and applicability: developers can model and test the performance of their application services in heterogeneous Cloud environments (Amazon EC2, Microsoft Azure) with little programming and deployment effort. CloudSim is an extensible simulation toolkit that enables modelling and simulation of Cloud Computing systems and application provisioning environments. CloudSim offers the following novel features [150]:

- Support for modelling and simulation of large-scale Cloud Computing environments, including data centres, on a single physical computing node.
- A self-contained platform for modelling Clouds, service brokers, provisioning, and allocation policies.
- Support for simulation of network connections among the simulated system elements.
- Availability of a virtualization engine that aids in the creation and management of multiple, independent, and co-hosted virtualized services on a data centre node.
- Flexibility to switch between space-shared and time-shared allocation of processing cores to virtualized services.

These compelling features of CloudSim would speed up the development of new application provisioning algorithms for Cloud Computing. Figure 6.1 shows the multi-layered design of the CloudSim software framework and its architectural components. The CloudSim simulation layer provides support for modelling and simulation of virtualized Cloud-based data centre environments including dedicated management interfaces for VMs, memory, storage, and bandwidth. The fundamental issues, such as provisioning of hosts to VMs, managing application execution, and monitoring dynamic system state, are handled by this layer. A Cloud provider, who wants to study the efficiency of different policies in allocating its hosts to VMs (VM

provisioning), would need to implement his strategies at this layer. Such implementation can be done by programmatically extending the core VM provisioning functionality. Cloud application developer can perform the following activities: (i) generate a mix of workload request distributions, application configurations; (ii) model Cloud availability scenarios and perform robust tests based on the custom configurations; and (iii) implement custom application provisioning techniques for Clouds and their federation.

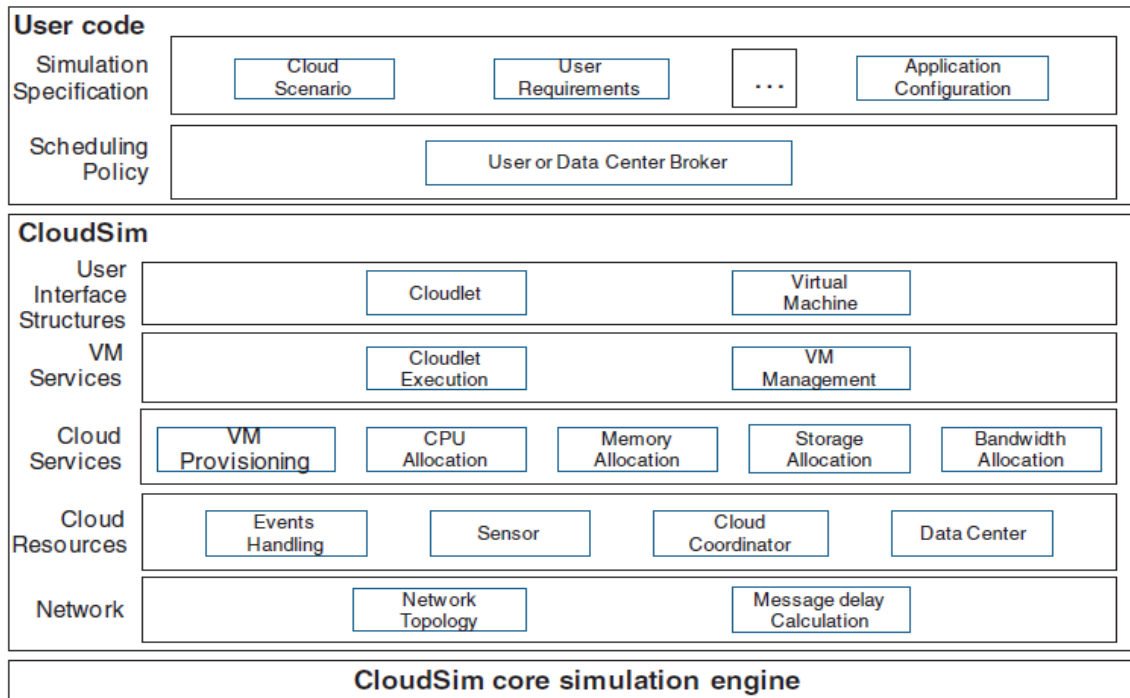


Figure 6.1: Multi-Layered Design of the CloudSim 3.0 [149]

CloudSim goal is to provide a generalized and extensible simulation framework that enables modelling, simulation, and experimentation of emerging Cloud Computing infrastructures and application services, allowing its users to focus on specific system design issues that they want to investigate, without getting concerned about the low level details related to Cloud-based infrastructures and services.

6.1.4 IntegratedNETJavaWeb

The tool is used integrate Java with Microsoft Technology is called IntegratedNETJavaWeb [150]. Through this tool some Java methods can call from .NET code, and pass some values to Java or .NET and vice versa. In this work, there is CWMP i.e. ASP.NET application, which interacts with Java programs i.e. CloudSim. The two IDE for Application Visual Studio 2010 and NetBeans IDE 7.1.2

have been used. This Figure 6.2 describes the main flow between Java and .NET. The main concept behind the scene is Applet. The JAR file is created which contains one applet and then Java methods called using applet from .NET. Once the data using JavaScript is retrieved, then it will use on the server side. A JAR file is created using Java.

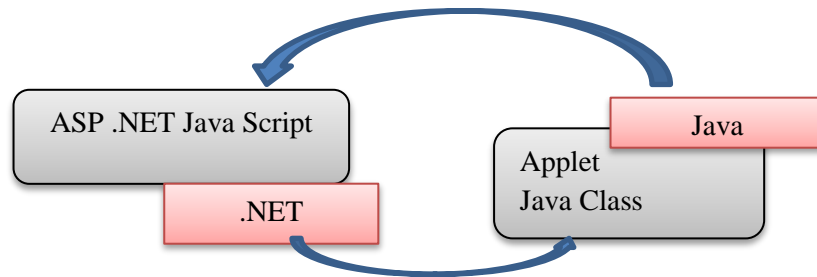


Figure 6.2: Integrate .NET and Java

For Java NetBeans is used. The implementation of IntegratedNETJavaWeb in ASP .NET as shown in Figure 6.3.

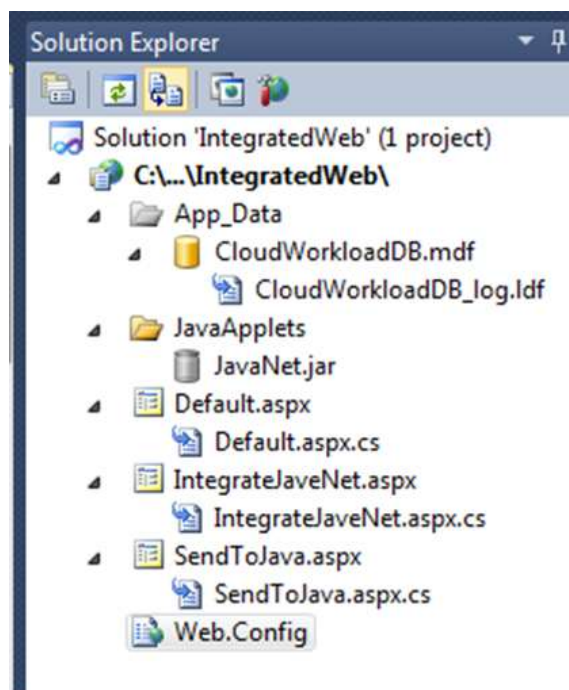


Figure 6.3: Implementation of IntegratedNETJavaWeb in ASP .NET

A “Jar” file, named JavaNet.Jar is created which contains appropriate methods. .NET is using to interact with NetBeans. The following is the Java Applet code. In the ASP.NET page, an Applet tag for invoking the applet is used then call any Java methods from file.

6.1.5 SQL Server 2008

Microsoft SQL Server 2008 is a relational database management system developed by Microsoft [151]. As a database, it is a software product whose primary function is to store and retrieve data as requested by other software applications, be it those on the same computer or those running on another computer across a network (including the Internet). There are at least a dozen different editions of Microsoft SQL Server aimed at different audiences and for different workloads (ranging from small applications that store and retrieve data on the same computer, to millions of users and computers that access huge amounts of data from the Internet at the same time).

6.2 Framework Execution

The framework executes the requests as follows:

1. In Cloud Workload Management Portal (CWMP), first of all the Cloud consumer tries to access the services through the user interface (website).
2. Cloud consumer must register before using the Cloud Workload Management Portal (CWMP).
3. After performing the task of the user's authentication and authorization, the home page of user will be displayed.
4. User writes their query regarding the Cloud workloads and then submits.
5. After this, Cloud consumer's workload details page will be displayed and a Cloud consumer selects the workload type, workload name, desired deadline, scheduling policy and enters the estimated budget.
6. Then CWMP will ask about confirmation. After confirmation the workload details will be submitted to CWMP. In queries, the workload displayed along with the workload id.
7. The Cloud provider manages all the workload queries after authentication and prepares the schedule by taking care of budget and desired deadline.
8. The number of registered users and number of queries will be displayed in the home page of Cloud provider.
9. The query and user details will be shown and Cloud provider can delete the user queries and edit or delete the user details.
10. Cloud provider selects the particular query for reply. Based on scheduling policy and availability of free slots the Cloud provider replies the queries by entering

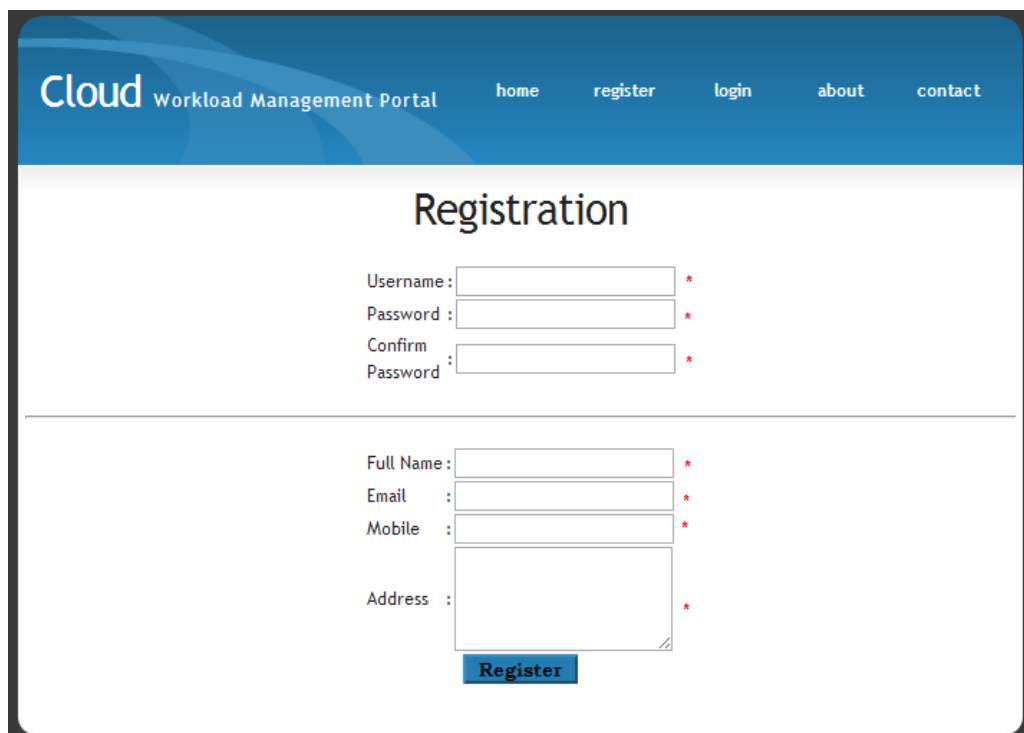
the schedule detail (Query reply, Workload name, Workload type, Start date, Start time, End date, End time and Estimated budget).

11. The workload schedule to a particular query will be displayed in the corresponding Cloud consumer account.
12. The user can update the password and contact to the Cloud provider through email regarding any technical query.

Thus, this framework exhibits how Cloud Workload Management can be done in a Cloud environment.

6.3 Implementation of Proposed Framework

Cloud Workload Management Portal has been developed to the optimized workload schedule based on scheduling policies in Microsoft Visual Studio 2010. CWMP is used to execute the Cloud workloads by using available resources within the available budget and desired deadline. The mapping of resources to the Cloud workloads is based on the scheduling policy. Its basic functionalities are shown as snapshots below. Figure 6.4 shows the registration form of the Cloud Workload Management Portal.



The screenshot shows the registration form of the Cloud Workload Management Portal. The page has a blue header with the text "Cloud Workload Management Portal" and navigation links for "home", "register", "login", "about", and "contact". The main content area is white and titled "Registration". It contains two sections of form fields. The first section includes "Username:", "Password:", and "Confirm Password:" fields, each followed by a red asterisk. The second section includes "Full Name:", "Email:", "Mobile:", and "Address:" fields, each followed by a red asterisk. A blue "Register" button is located at the bottom of the form.

Figure 6.4: Registration Form

Cloud consumer enters the registration details includes username, password, confirm password, full name, email, mobile and address. The validation is used for each text

box denoted by Asterisk (*). The purpose of this is to show that every field is mandatory to fill. None of the field must be leaved unfilled. If so, then registration will fail. So for the successful registration new user must provide all the required information. New user presses the register button after filling all the fields. As for the case when new user leave any of the required field empty as shown in Figure 6.5.

The screenshot shows a web page titled "Cloud Workload Management Portal" with a navigation menu (home, register, login, about, contact). The main heading is "Registration". The form contains the following fields:

- Username: sukhpalsingh
- Password: [masked]
- Confirm Password: [masked]
- Full Name: [empty]
- Email: [empty]
- Mobile: [empty] with a red asterisk (*) to its right
- Address: [empty]

A blue "Register" button is located at the bottom of the form.

Figure 6.5: Registration Failed

After entering all the required details for successful registration new user will click on “registration” button as shown in Figure 6.6.

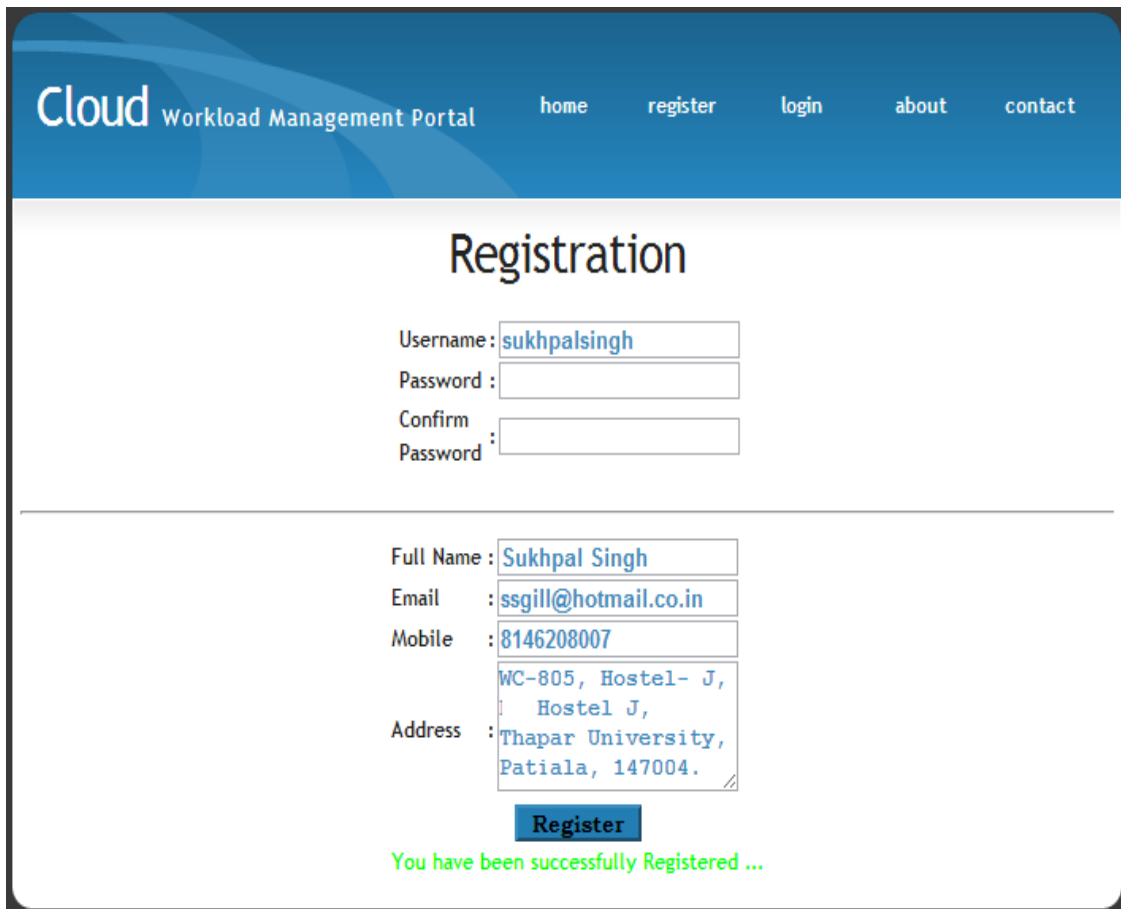
The screenshot shows the same "Registration" page as Figure 6.5, but with the form fields filled out. The fields contain the following information:

- Username: sukhpalsingh
- Password: [masked]
- Confirm Password: [masked]
- Full Name: Sukhpal Singh
- Email: ssgill@hotmail.co.in
- Mobile: 8146208007
- Address: WC-805, Hostel- J, Hostel J, Thapar University, Patiala, 147004.

The blue "Register" button is still present at the bottom.

Figure 6.6: New User Details

The message “You have been successfully registered ...” will be displayed on the registration page as shown in Figure 6.7.



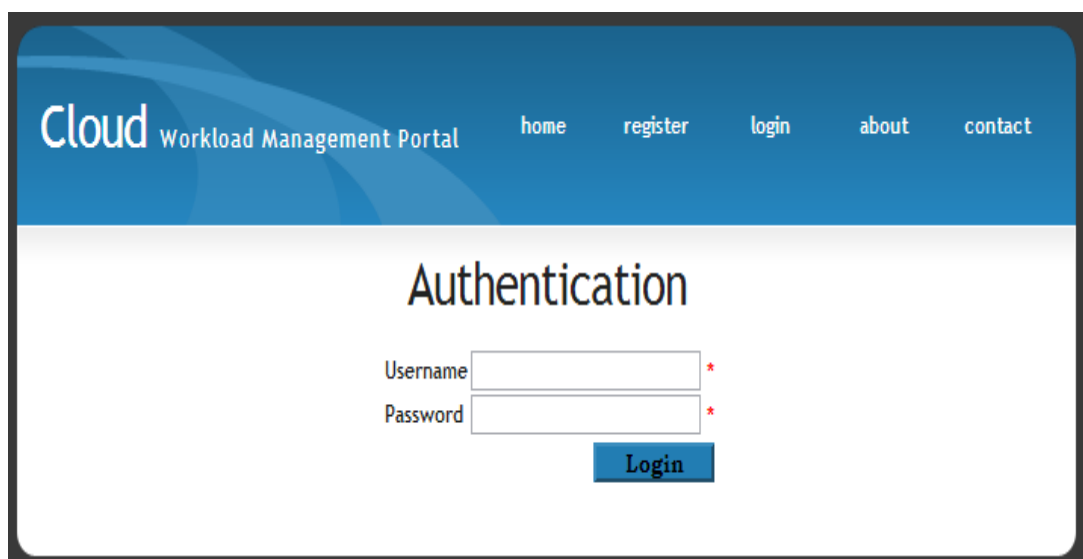
The screenshot shows the registration page of the Cloud Workload Management Portal. The header includes the logo and navigation links: home, register, login, about, and contact. The main heading is "Registration". The form contains the following fields and values:

- Username:
- Password:
- Confirm Password:
- Full Name:
- Email:
- Mobile:
- Address:

A blue "Register" button is located below the address field. Below the button, a green message reads: "You have been successfully Registered ...".

Figure 6.7: Successful Registration

After registration, user can click on “login” to proceed further as shown in Figure 6.8.



The screenshot shows the login page of the Cloud Workload Management Portal. The header includes the logo and navigation links: home, register, login, about, and contact. The main heading is "Authentication". The form contains the following fields and values:

- Username:
- Password:

Red asterisks (*) are placed to the right of both input fields. A blue "Login" button is located below the password field.

Figure 6.8: Login Page

The purpose of Asterisk (*) is to show that username and password is mandatory to fill. None of the field must be left unfilled. If so, then login will fail. User presses the login button after filling both the fields. As for the case when user enters wrong username and/or password as shown in Figure 6.9.

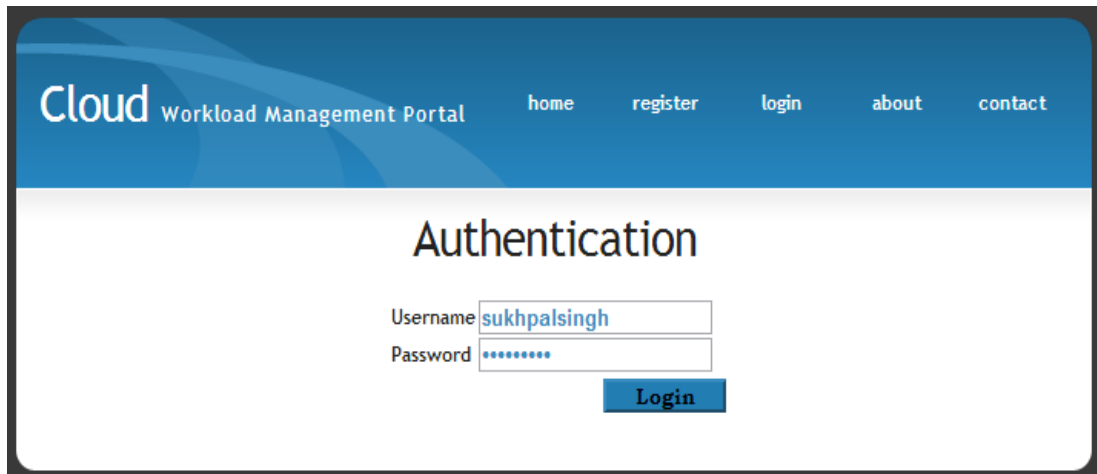


Figure 6.9: Wrong Username and/or Password

After entering wrong username and/or password the Cloud Workload Management Portal (CWMP) display the message “Invalid username/Password” as shown in Figure 6.10.

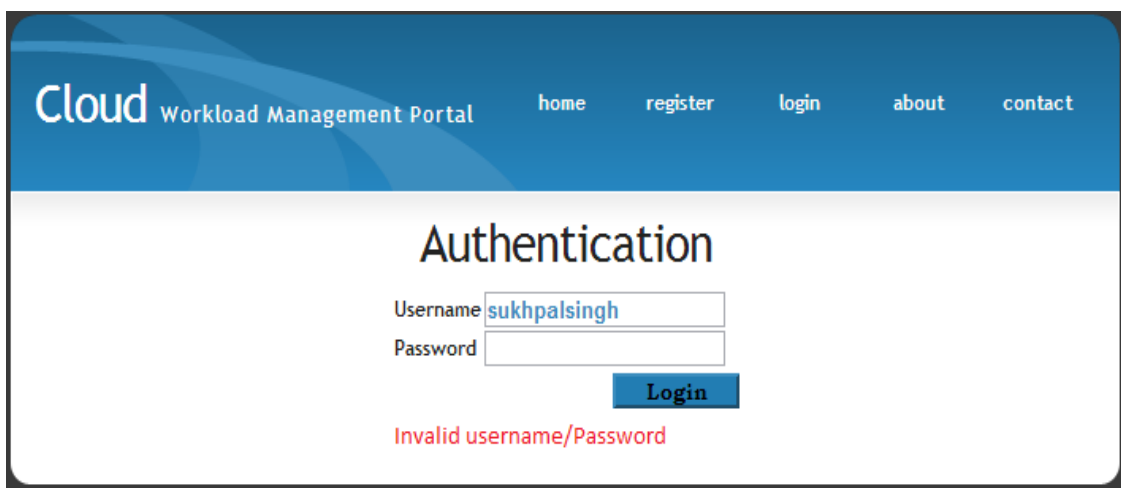


Figure 6.10: Login Unsuccessful

Figure 6.11 shows the correct username and password entered by Cloud user in the CWMP. The password is case sensitive, user will be taking care of these constraints while logging to the Cloud Workload Management Portal.

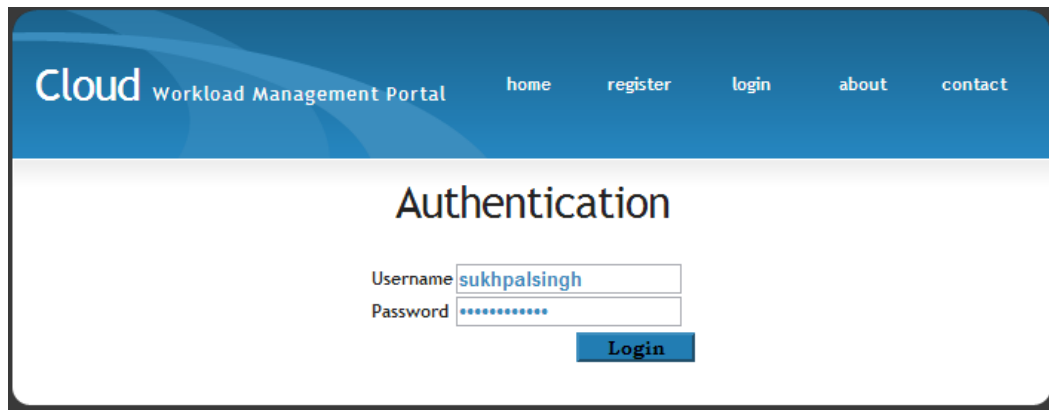


Figure 6.11: Correct Login Detail

Once user successfully logged in the Cloud Workload Management Portal (CWMP), the user home page will be displayed as shown in Figure 6.12. There are four options available on the user home page i.e. home, queries, password (to change password) and logout. To continue with this system Cloud consumer will select the workload name, workload type, desired deadline and preferred policy from the drop down list and entering the estimated budget (minimum \$50). The workload types are Compute, Communication, Administrator and Storage.

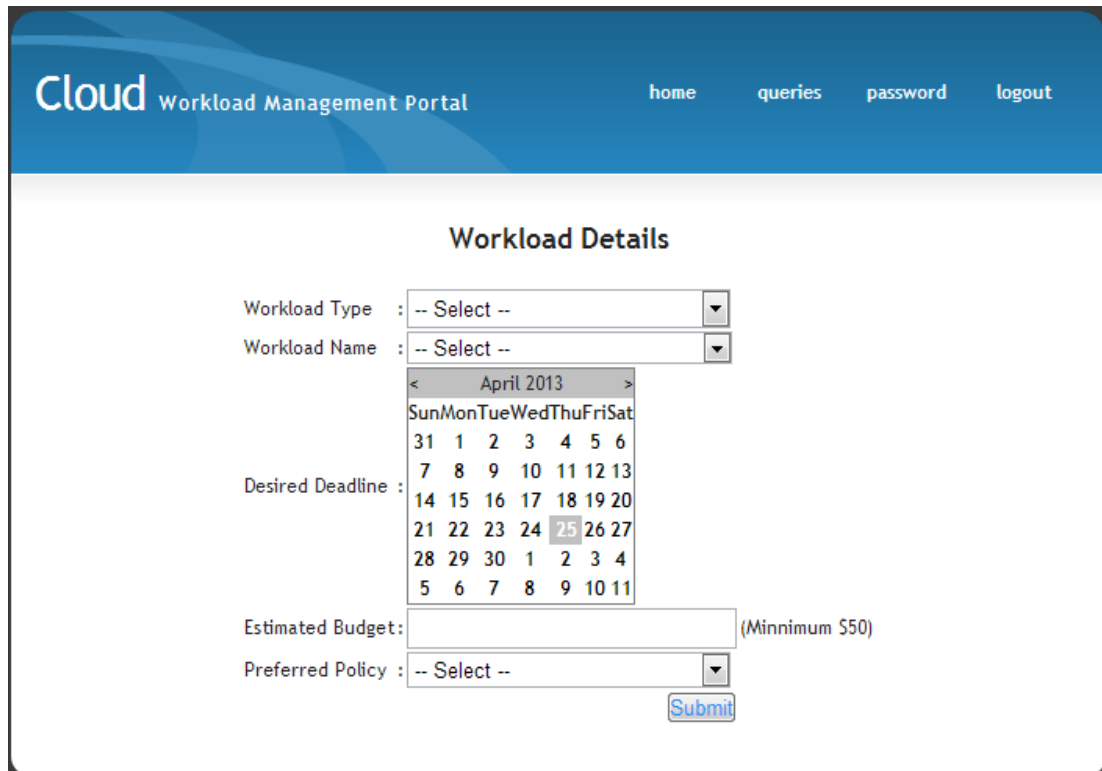


Figure 6.12: Workload Details

The available workload names are displayed in Figure 6.13 in the drop down list.

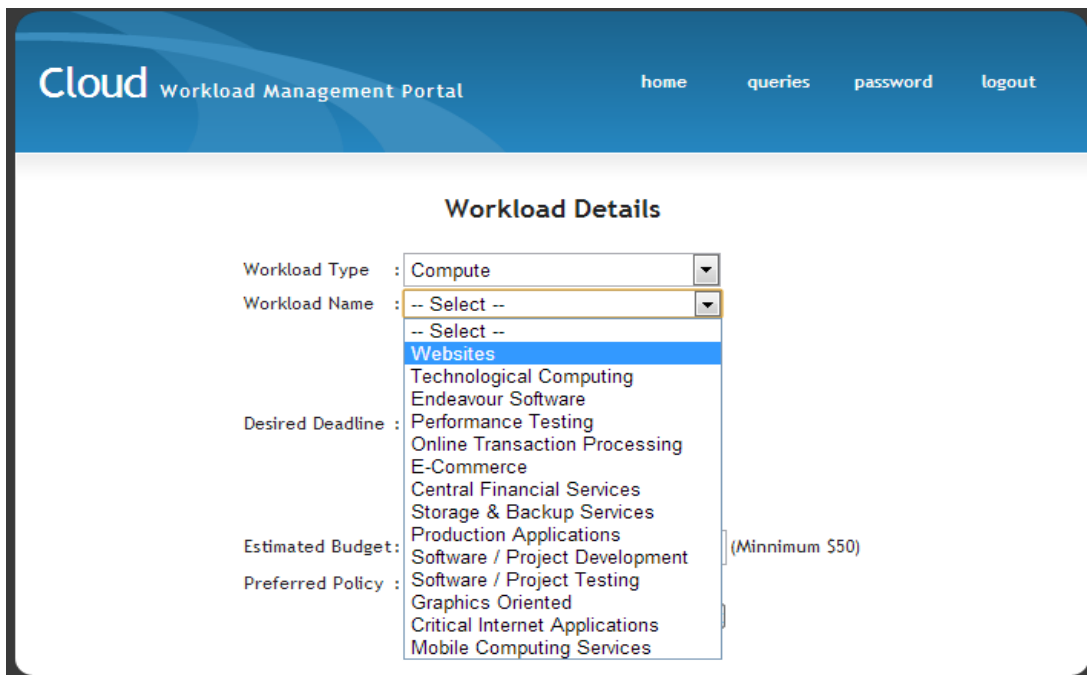


Figure 6.13: Select Workload Name

After selecting the “workload type” (Compute) and “workload name” (Websites), user selects “desired deadline” and “preferred policy” and enter estimated budget as shown in Figure 6.14.

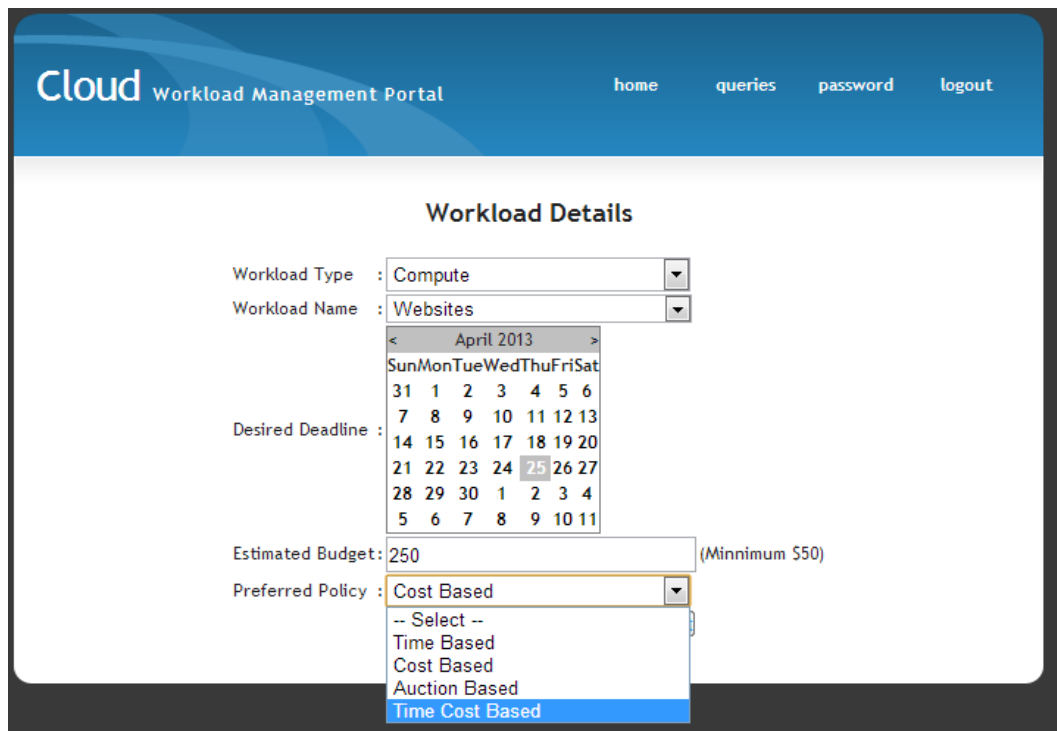


Figure 6.14: Select Preferred Policy

The complete workload details entered by user are shown in Figure 6.15.

Cloud Workload Management Portal home queries password logout

Workload Details

Workload Type : Compute

Workload Name : Websites

Desired Deadline :

< April 2013 >						
Sun	Mon	Tue	Wed	Thu	Fri	Sat
31	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	1	2	3	4
5	6	7	8	9	10	11

Estimated Budget: 250 (Minimum \$50)

Preferred Policy : Cost Based

Submit

Figure 6.15: Entered Workload Details

After entering details, CWMP will ask confirmation. After click on “Yes” button the workload details will be confirmed and submitted to the CWMP as shown in Figure 6.16.

Cloud Workload Management Portal home queries password logout

Workload Details

Workload Type : Compute

Workload Name : Websites

Desired Deadline : 4/25/2013 12:00:00 AM

Estimated Budget : 250

Preferred Policy : Time Cost Based

Is this information correct ?

Yes No

Figure 6.16: Confirmation of Workload Details

All the workload details entered by a particular user are displayed in user account by clicking on “queries” as shown in Figure 6.17.

		id	WorkloadType	WorkloadName	DesiredDeadline
Delete	Select	17	Storage	Critical Internet Applications	4/24/2013 12:00:00 AM
Delete	Select	16	Administration	Graphics Oriented	4/17/2013 12:00:00 AM
Delete	Select	14	Storage	Storage & Backup Services	4/27/2013 12:00:00 AM
Delete	Select	12	Compute	Websites	4/25/2013 12:00:00 AM

Figure 6.17: User Queries

All the Cloud workload details submitted by the user are stored in Cloud provider home page. Cloud provider enters the login details to generate the Cloud workload as shown in Figure 6.18.

Cloud Workload Management Portal

home register login about contact

Authentication

username

password

Figure 6.18: Cloud Provider Login

After successfully logged in, the number of registered users and numbers of queries displayed on the home page of Cloud provider are displayed. There are three other options i.e. queries (to show user queries regarding Cloud workload), users (to show

the details of registered users) and reply (to reply a particular user about Cloud workload schedule) and finally Cloud provider can logged out after generating all the workload schedules as shown in Figure 6.19.

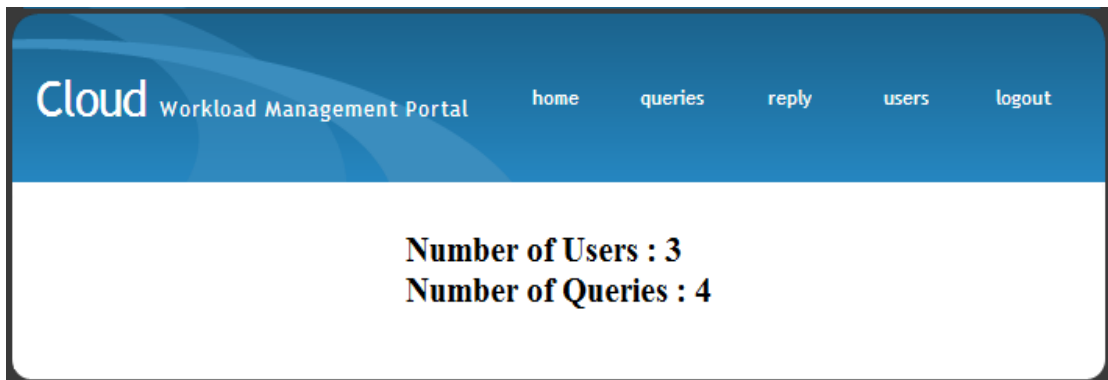


Figure 6.19: Cloud Provider Home Page

All user queries regarding the Cloud workloads are displayed in the tabular form as shown in Figure 6.20 and Cloud provider can delete the query for which Cloud workload is executed successfully.

	id	username	WorkloadType	WorkloadName	DesiredDeadline	EstimatedBudget	PreferredPolicy
Delete	4	mark	Compute	Endeavour Software	3/5/2013 12:00:00 AM	100	Cost Based
Delete	5	Pankaj	Compute	Websites	3/11/2013 12:00:00 AM	100	Time Based
Delete	9	sukhpalsingh	Compute	Websites	4/10/2013 12:00:00 AM	200	Time Based
Delete	12	SukhpalSingh	Compute	Websites	4/25/2013 12:00:00 AM	250	Time Cost Based

Figure 6.20: User Queries

The list of registered users is displayed in tabular form as shown in Figure 6.21. Cloud provider can edit or delete the user.

	Username	FullName	Org	Email	Mobile	Address
Edit Delete	mark	mark Cheema	markTech	markcheema@gmail.com	9855886588	Patiala
Edit Delete	Pankaj	Pankaj Nautiyal	Thapar University	pankaj.leon@gmail.com	8968311533	Patiala
Edit Delete	SukhpalSingh	Sukhpal Singh	Thapar University	ssgill@hotmail.co.in	8146208007	WC-805 Hostel J TU Patiala.

Figure 6.21: List of Registered Users

The Cloud provider reply the user queries by click on “select” link and generate the workload schedule based on the scheduling policy as shown in Figure 6.22.

	id	username	WorkloadType	WorkloadName	DesiredDeadline	EstimatedBudget	PreferredPolicy
Select	4	mark	Compute	Endeavour Software	3/5/2013 12:00:00 AM	100	Cost Based
Select	5	Pankaj	Compute	Websites	3/11/2013 12:00:00 AM	100	Time Based
Select	9	sukhpalsingh	Compute	Websites	4/10/2013 12:00:00 AM	200	Time Based
Select	12	SukhpalSingh	Compute	Websites	4/25/2013 12:00:00 AM	250	Time Cost Based

Figure 6.22: Reply to User Queries

In Figure 6.23, the Cloud workload schedule is generating for the workload id 12.

The screenshot shows the 'Reply' form in the Cloud Workload Management Portal. The form is titled 'Reply' and is located in the center of the page. The form contains the following fields:

Comments	:	<input type="text"/>
Workload Type	:	-- Select --
Workload Name	:	-- Select --
Start Date	:	Storage
Start Time	:	Communication
End Date	:	Administration
End Time	:	<input type="text"/>
Estimated Budget	:	<input type="text"/>

At the bottom of the form, there is a 'Reply' button.

Figure 6.23: Generating Workload Schedule

Based on user details, the details of generated Cloud workload schedule is entered and click the reply button to send these details to specific user as shown in Figure 6.24.

The screenshot shows the 'Reply' form in the Cloud Workload Management Portal with the following data entered:

Comments	:	Schedule Ready
Workload Type	:	Compute
Workload Name	:	Websites
Start Date	:	24-04-2013
Start Time	:	10:00
End Date	:	24-04-2013
End Time	:	21:00
Estimated Budget	:	230

At the bottom of the form, there is a 'Reply' button.

Figure 6.24: Generated Workload Schedule

The workload schedule generated by Cloud provider is displayed in the specific user account as shown in Figure 6.25.

Cloud Workload Management Portal		home	queries	password	logout
Reply to the Query is :					
Query	Schedule Ready				
Workload Type	Compute				
WorkloadName	Websites				
Start Date	4/24/2013				
End Date	4/24/2013				
Start Time	10:00				
End Time	21:00				
Estimated Budget	230				
Reply Date	4/19/2013 12:00:00 AM				

Figure 6.25: Final Workload Schedule

6.4 Results of Clustering of Cloud Workloads

An ARFF (Attribute-Relation File Format) file is an ASCII text file that describes a list of workloads sharing a set of clusters. An Attribute-Relation File Format file (k.means.arff) for Cloud workloads clusters and their corresponding values are shown in Figure 6.26.

```
@relation K.means
@attribute Compute {6,0,11,3,9,10,7,5}
@attribute Storage {3,8,0,6,7,4,2}
@attribute Communication {0.15,9,5.1,5.9,2.9,3.1,4.1,1.1,0.1}
@attribute Administration {3.7,9.7,1.3,6.7,0.7,0.3,2.7,4.7}
@attribute NearestCluster {C3,C1,C4,C2}

@data
6,3,0.1,3.7,C3
0,3,5.9,9.7,C1
11,8,5.1,1.3,C4
0,3,5.9,9.7,C1
11,8,5.1,1.3,C4
3,0,2.9,6.7,C2
9,6,3.1,0.7,C4
3,0,2.9,6.7,C2
9,6,3.1,0.7,C4
9,6,3.1,0.7,C4
10,7,4.1,0.3,C4
7,4,1.1,2.7,C3
5,2,0.1,4.7,C3
```

Figure 6.26: Attribute-Relation File Format File (k.means.arff)

ARFF files have two distinct sections. The first section is the Cluster information, which is followed the workload distance information. The Header of the ARFF file contains the name of the relation, a list of the clusters (named Compute, Storage, Communication and Administration), and their minimum distance value. These results are measured with the help of data mining tool i.e. WEKA Tool [152]. The results are taken through the use of WEKA tool. There are four different clusters (C1 = Compute, C2 = Storage, C3 = Communication and C4 = Administration) along with nearest cluster. In this algorithm, the minimum distance is calculated and put a particular workload into suitable cluster. The scattering of workloads amongst four clusters in the form of plot matrix is shown in Figure 6.27.

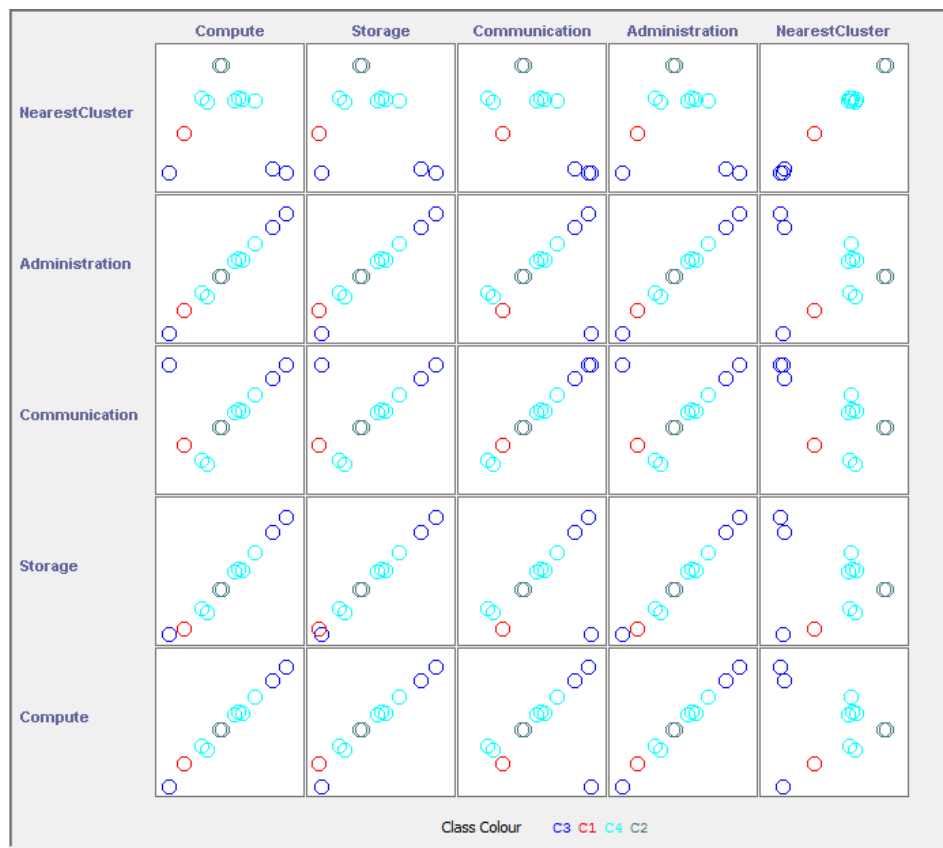


Figure 6.27: Scattering of Workloads amongst Four Clusters

6.5 Experimental Results

3000 independent Cloud workloads created randomly. Here, 10 different Cloud workloads along with identification number (WId) are considering to show how proposed four scheduling policies working in different criteria and perform better than already existed algorithms. The attributes of the Cloud workload include deadline, estimated budget and policy. Table 6.1 describes the Cloud workload details.

Table 6.1: Cloud Workload Details

WId	Workload Name	Workload Type	W_d (H)	B_E (\$)	D_t (H)	E_t (H)
W1	Web sites	Communication	12:00	100	12	12
W2	Technological computing	Compute	4:00	62	4	6.45
W3	Endeavour software	Administration	6:00	120	6	5
W4	Performance testing	Compute	21:00	170	21	12.35
W5	Online transaction processing	Administration	10:00	155	10	6.45
W6	E-Com	Storage	2:00	200	2	1
W7	Central financial services	Administration	4:00	252	4	1.6
W8	Storage and backup services	Storage	20:00	265	20	7.54
W9	Productivity applications	Administration	4:00	72	4	5.55
W10	Software/Project development and testing	Administration	14:00	65	14	21.53

Where W_d = Desired Deadline in Hours, B_E = Estimated Budget in Dollars, D_t = Deadline Time in Hours and E_t = Estimated Time in Hours. In order to evaluate the effectiveness of the scheduling policies discussed in chapter 5, a simulator namely CloudSim 3.0 has been used to calculate the execution time for each of them. In this section, the performance of all the algorithms is discussing. Table 6.2 shows the characteristics of resources and Cloudlets that has been used for all the experiments. User Cloud workloads as independent parallel applications are modelled which is computation intensive. Thus the data dependency among the Cloud workloads in the parallel applications is negligible. Each Cloud workload is parallel and is hence considered to be independent of any other Cloud workload.

Table 6.2: Scheduling Parameters and their Values

Parameter	Value
Number of resources	50-250
Number of Cloudlets (Workloads)	3000
Bandwidth	1000 - 3000 B/S
Size of Cloud Workload	10000+ (10%–30%) MB
Number of PEs per machine	1
PE ratings	100-4000 MIPS
Cost per Cloud Workload	\$3–\$5
Memory Size	2048-12576 MB
File size	300 + (15%–40%) MB
Cloud Workload output size	300 + (15%–50%) MB

All the policies consider only one dimensional QoS (processing speed) described in Table 6.3. The QoS is generated to be 1 to 32 with the ratio following given distribution of the QoS request.

Table 6.3: Cost Related to Different Processing Speed

Service	Processing Speed (PS)	MIPS Rating	Cost (\$)/Workload
Service 1	1	100	0
Service 2	2	250	50
Service 3	4	500	100
Service 4	8	1000	200
Service 5	16	2000	400
Service 6	32	4000	800

The Minimum Execution Time (E_t) is calculated with the formula given below:

$$E_t = \frac{D_t}{B_E \times PS} \times 100$$

Where E_t = Minimum Execution Time, D_t = Deadline Time, B_E = Estimated Budget and PS = Processing Speed.

6.5.1 First Come First Serve (FCFS) Based Scheduling Policy

Allocate Resources FCFS based without taking care of effective mapping (without considering execution time and execution cost). By default, PS=1.

E_t	12	6.45	5	12.35	6.45	1	1.6	7.54	5.55	21.53
workload	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10
Resource	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10

6.5.2 Compromised Cost - Time Based (CCTB) Scheduling Policy

Deadline urgency: Deadline urgency (D_u), which specifies Cloud customer urgency to get workload (s) completed, is defined as:

$$D_u = \frac{D_t}{E_t} - 1$$

Where D_u = Deadline Urgency, D_t = Deadline Time and E_t = Minimum Execution Time. This metric shows how the scheduler deals with Cloud customer with different requirement on time. The value of Communication Cost (C_c) and Communication Time (C_t) depends on the Deadline Urgency (D_u) as shown in Table 6.4.

Table 6.4: Deadline Urgency

D_u	C_c (\$)	C_t (H)
$D_u > 0.75$	1	10/60 = 0.17
$0.25 \leq D_u \leq 0.75$	3	60/60 = 1
$D_u < 0.25$	5	120/60 = 2

The Total Expected Cost (TEC) and Total Expected Completion Time (TECT) are calculated using the formula given below and the value for each workload will be calculated as shown in Table 6.5.

$$TECT = C_t + E_t$$

$$TEC = C_c + C_{min}$$

Where, C_t = Communication Time, E_t = Minimum Execution Time, C_c = Communication Cost and C_{min} = Minimum cost of a particular workload. For execution of Cloud workload under this policy the condition will be fulfilled strictly:

$$TECT \leq D_t \ \&\& \ TEC \leq B_E$$

Table 6.5: Total Expected Cost and Total Expected Completion Time

WId	W_d (H)	B_E (\$)	D_t (H)	E_t	C_t (H)	C_c (\$)	D_u	TEC	TECT
W1	12:00	100	12	12	2	5	0	105	14
W2	4:00	62	4	6.45	2	5	-ve	67	8.7
W3	6:00	120	6	5	2	5	0.2	125	7
W4	21:00	170	21	12.35	1	3	0.7	173	13.35
W5	10:00	155	10	6.45	1	3	0.55	158	7.7
W6	2:00	200	2	1	0.17	1	1	201	1.17
W7	4:00	252	4	1.58	0.17	1	1.53	253	2.6
W8	20:00	265	20	7.54	0.17	1	1.65	263	8.69
W9	4:00	72	4	5.55	2	5	-ve	75	7.71
W10	14:00	65	14	21.53	2	5	-ve	70	23.53

$$\text{Time Difference } (T_d) = D_t - \text{TECT}$$

Where W_d = Desired Deadline in Hours, B_E = Estimated Budget in Dollars, D_t = Deadline Time in Hours, C_t = Communication Time, E_t = Minimum Execution Time, C_c = Communication Cost, TECT = Total Expected Completion Time, D_u = Deadline Urgency and TEC = Total Expected Cost. The Time Difference (T_d) is calculated by the formula defined above as shown in Table 6.6. If the value of $T_d > 0$

then the Cloud workload will be executed before their deadline otherwise it will not be executed before deadline.

Table 6.6: Time Difference (T_d)

WId	B_E (\$)	D_t (H)	TEC	TECT	T_d	Deadline Fulfilled
W1	100	12	105	14	-2	No
W2	60	4	65	8.7	-4.7	No
W3	120	6	125	7	-1	No
W4	170	21	173	13.35	7.65	Yes
W5	150	10	153	7.7	2.3	Yes
W6	200	2	201	1.17	0.83	Yes
W7	250	4	253	2.6	1.4	Yes
W8	260	20	263	8.69	11.4	Yes
W9	70	4	75	7.71	-3.71	No
W10	65	14	70	23.53	-9.53	No

Number of Deadlines Missed: 5

6.5.2.1 Rescheduling of Cloud Workloads

The workloads W4, W5, W6, W7 and W8 have extra time than required for execution within their deadline, the total extra time $(7.65+2.3+0.83+1.4+11.4) = 23.58$ and the time required to execute the pending workloads $(-2-4.7-1-3.71-9.53) = -20.94$ is less than the time available. So the W1, W2, W3, W9 and W10 will be executed within their deadline and budget respectively as shown in Table 6.7.

Table 6.7: Rescheduling of Cloud Workloads

WId	D_t (H)	TECT	Updated D_t (H)	Deadline Fulfilled
W1	12	14	14	Yes
W2	4	8.7	8.7	Yes
W3	6	7	7	Yes
W4	21	13.35	13.35	Yes
W5	10	7.7	7.7	Yes
W6	2	1.17	1.17	Yes
W7	4	2.6	2.6	Yes
W8	20	8.69	8.69	Yes
W9	4	7.71	7.71	Yes
W10	14	23.53	23.53	Yes

After Rescheduling, Deadline Missed = 0

The implementation of Compromised Cost-Time Based Scheduling Policy follows the algorithm shown in Figure 5.4. Calculate the TEC, TECT and Time Difference (T_d) to allocate the resources. The allocation agent find the missed deadlines and calculate Time Difference for each workload then uses the extra available time to the workloads with missed deadlines and execute all the Cloud workloads within their corresponding deadlines. The performance evaluation criterion to evaluate the performance of Compromised Cost - Time Based (CCTB) for resource scheduling has been defined. The two parameters namely cost and execution time has been selected. The cost and execution time measured in dollars and seconds respectively. To validate CCTB algorithm, 3000 Cloud workloads and 120-140 resources are considered. In order to evaluate the performance of CCTB, the effects of different number of Cloud workloads has been investigated. The two existing reference algorithms namely CTC [153] and DBD-CTO [79] have been used.

- Compromised Time Cost (CTC) Scheduling Algorithm – Compromised Cost - Time scheduling policy considers the characteristics of Cloud Computing to accommodate instance-intensive cost-constrained workflows by compromising execution time and cost. The simulation performed demonstrates that the algorithm can cut down the mean execution cost and shorten the mean execution time within the user-designated execution cost [153].
- Deadline and Budget Distribution-Based Cost-Time Optimization (DBD-CTO) workflow scheduling algorithm that minimizes execution cost while meeting timeframe for delivering results and analyse the behaviour of the algorithm. In this algorithm, the two constraints are considered: deadline and budget. For the workflow, a list of three services for each task of the workflow was created. The scheduler that is implemented in the broker part calls DBD-CTO to choose a particular service such that overall workflow execution should be in deadline and budget constraints specified by the user [79].

Both the algorithms are doing resource scheduling based on the homogenous Cloud workloads. The concept of rescheduling is also not used in these existed algorithms. All the three algorithms Compromised Time Cost (CTC) Scheduling Algorithm (existing), Deadline and Budget Distribution-Based Cost-Time Optimization (DBD-CTO) (existing) and Compromised Cost Time Based (CCTB) (proposed) have been

implemented in CloudSim 3.0 by making the changes in the VMScheduler.java and then compared in different scenarios.

Test Case 1: Execution Time Comparison of Cloud Workloads

After executing the values on scheduling parameters for different algorithms, the Cloud simulator CloudSim provides the execution summary for DBD-CTO shown in Figure 6.28. The execution time for executing the same Cloud workloads by using same resources is 1803 seconds.

```
Datacenter_0 is starting...
Broker is starting...
Entities started.
0.0: Broker: Cloud Resource List received with 1 resource(s)
0.0: Broker: Trying to Create VM #0 in Datacenter_0
0.0: Broker: VM #0 has been created in Datacenter #2, Host #0
0.0: Broker: Sending cloudlet 0 to VM #0
1803.0: Broker: Cloudlet 0 received
1803.0: Broker: All cloudlets executed. Finishing...
1803.0: Broker: Destroying VM #0
Broker is shutting down...
Simulation: No more future events
CloudInformationService: Notify all Cloudsim entities for shutting down.
Datacenter_0 is shutting down...
Broker is shutting down...
Simulation completed.
Simulation completed.
```

```
===== OUTPUT =====
Cloudlet ID   STATUS   Data center ID   VM ID   Time   Start Time   Finish Time
0            SUCCESS   2                0       1803   0.0          1803
```

Figure 6.28: Execution of DBD-CTO in CloudSim

The CTC algorithm is executed with same resources in the CloudSim as shown in Figure 6.29. The time taken to execute the same Cloud workloads by CTC algorithm is 1330 seconds.

```
Datacenter_0 is starting...
Broker is starting...
Entities started.
0.0: Broker: Cloud Resource List received with 1 resource(s)
0.0: Broker: Trying to Create VM #0 in Datacenter_0
0.0: Broker: VM #0 has been created in Datacenter #2, Host #0
0.0: Broker: Sending cloudlet 0 to VM #0
1330.0: Broker: Cloudlet 0 received
1330.0: Broker: All cloudlets executed. Finishing...
1330.0: Broker: Destroying VM #0
Broker is shutting down...
Simulation: No more future events
CloudInformationService: Notify all cloudsim entities for shutting down.
Datacenter_0 is shutting down...
Broker is shutting down...
Simulation completed.
Simulation completed.
```

```
===== OUTPUT =====
Cloudlet ID   STATUS   Data center ID   VM ID   Time   Start Time   Finish Time
0            SUCCESS   2                0       1330   0.0          1330
```

Figure 6.29: Execution of CTC in CloudSim

CCTB is implemented and executed with same environment. The same Cloud workload is executed in this algorithm in 1243 seconds. The execution time taken by

CCTB algorithm is lesser than other scheduling algorithms. The execution summary of CCTB in CloudSim is shown in Figure 6.30.

```
Datacenter_0 is starting...
Broker is starting...
Entities started.
0.0: Broker: Cloud Resource List received with 1 resource(s)
0.0: Broker: Trying to Create VM #0 in Datacenter_0
0.0: Broker: VM #0 has been created in Datacenter #2, Host #0
0.0: Broker: Sending cloudlet 0 to VM #0
1243.0: Broker: Cloudlet 0 received
1243.0: Broker: All cloudlets executed. Finishing...
1243.0: Broker: Destroying VM #0
Broker is shutting down...
Simulation: No more future events
CloudInformationService: Notify all cloudsim entities for shutting down.
Datacenter_0 is shutting down...
Broker is shutting down...
Simulation completed.
Simulation completed.
```

Cloudlet ID	STATUS	Data center ID	VM ID	Time	Start Time	Finish Time
0	SUCCESS	2	0	1243	0.0	1243

Figure 6.30: Execution of CCTB in CloudSim

Figure 6.31 demonstrates the effectiveness of the CCTB algorithm in managing the time requirement of the Cloud user. The characteristics of Cloudlets are used to compare the execution time of three algorithms. In this case lowest execution time was achieved in case of CCTB algorithm whereas DBD-CTO resulted in the highest execution time.

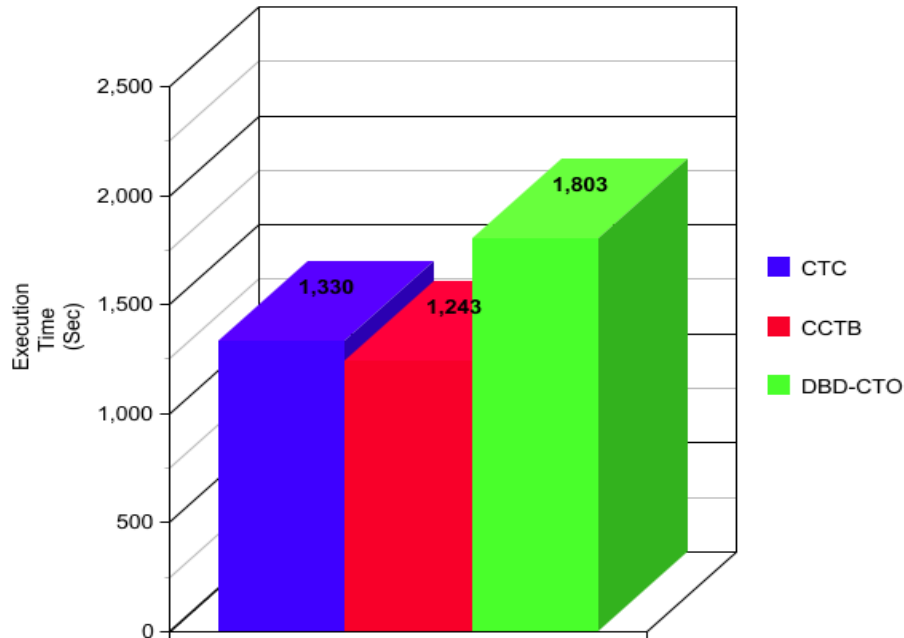


Figure 6.31: Execution Time Comparison of Cloud Workloads

Test Case 2: Cost Comparison of Cloud Workloads

Figure 6.32 shows the effect on cost by three algorithms. Figure shows the lowest cost was achieved by CCTB where DBD-CTO resulted in highest cost.

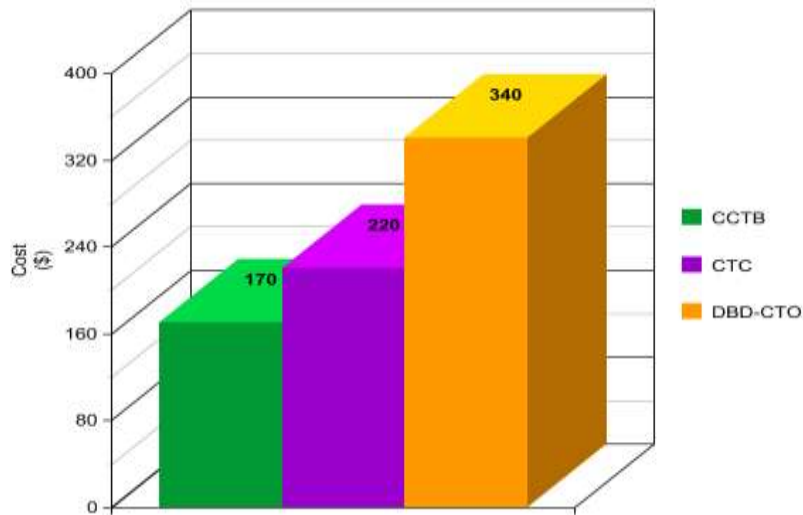


Figure 6.32: Cost Comparison of Cloud Workloads

Test Case 3: Execution Time for Different Number of Resources

Figure 6.33 shows the effect of increasing the number of resources, while keeping the number of Cloud workloads being executed. In this experiment, 3000 Cloud workloads were executed with varying numbers of resources. The results depict that by increasing the number of resources, the execution time decreases. CCTB performs better than DBD-CTO and CTC.

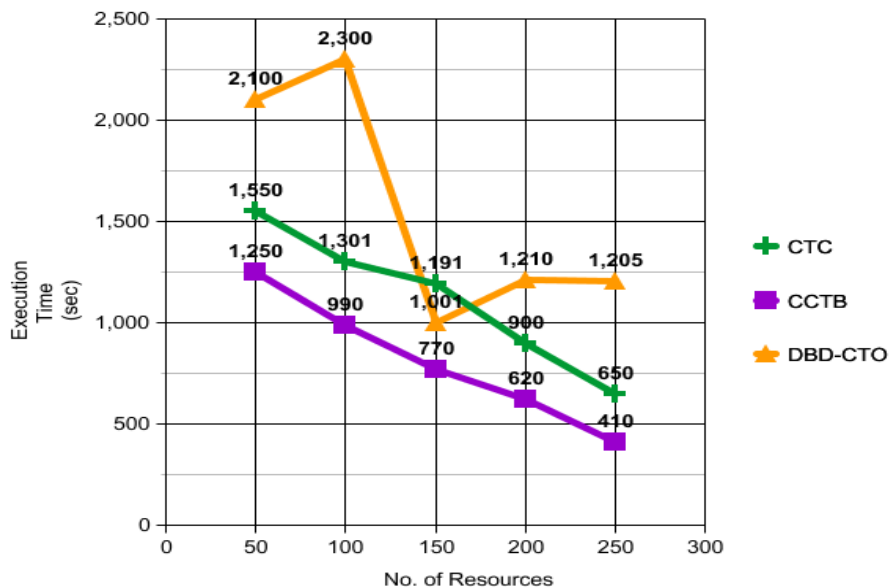


Figure 6.33: Execution Time for Different Number of Resources

Test Case 4: Cost for Different Number of Resources

The cost of execution of different Cloud workloads for three algorithms varies. The costs of resources are decreasing with increasing the number of resources. Figure

6.34 shows the CCTB algorithm executes the same number of Cloud workloads at a minimum cost.

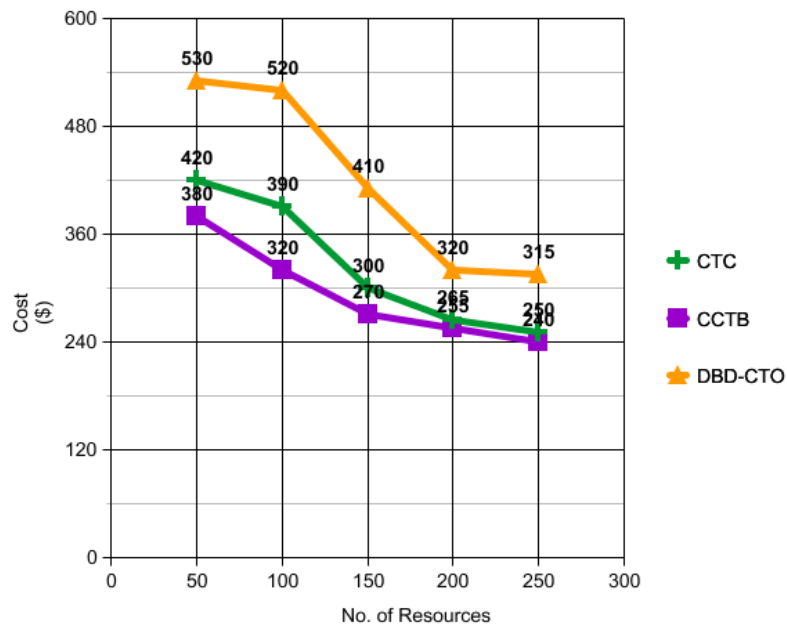


Figure 6.34: Cost for Different Number of Resources

Test Case 5: Execution Time for Different Number of Cloud Workloads

The execution time is increasing with the increase in number of Cloud workloads and the execution time of CCTB for same number of Cloud workloads is slightly lesser than CTC as shown in Figure 6.35.

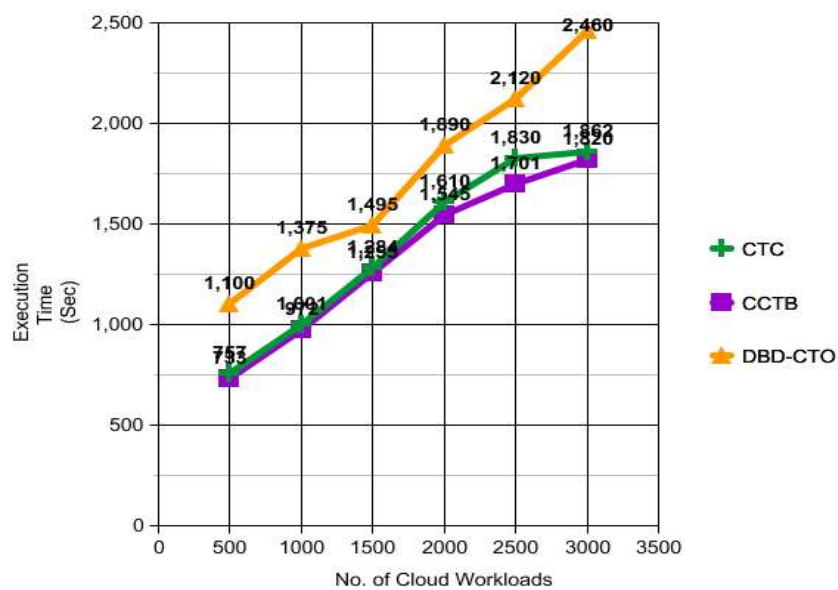


Figure 6.35: Execution Time for Different Number of Cloud Workloads

Test Case 6: Cost for Different Number of Cloud Workloads

Figure 6.36 shows that cost per Cloud workload increases as the number of submitted Cloud workload increases.

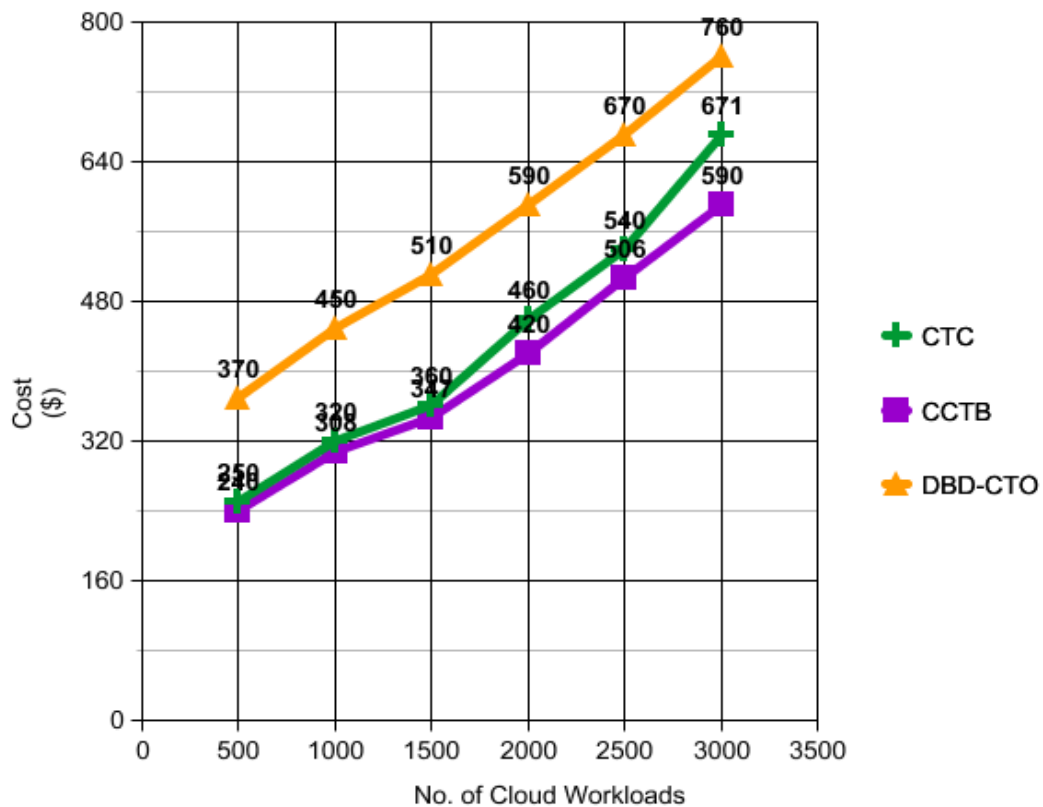


Figure 6.36: Cost for Different Number of Cloud Workloads

The existing algorithm based workload's execution resulted in a schedule which is expensive in comparison to the CCTB algorithm. From all the experimental results, the workload execution using the CCTB algorithm performs better. The overall cost for Cloud consumer's workload execution is less.

Test Case 7: Number of Missed Deadlines

There are different numbers of deadlines missed in different algorithms. With increasing the number of Cloud workloads, the number of deadlines missed is also increasing. The number of deadlines missed in Deadline and Budget Distribution-Based Cost-Time Optimization (DBD-CTO) are maximum and minimum in Compromised Cost Time Based (CCTB) as shown in Figure 6.37.

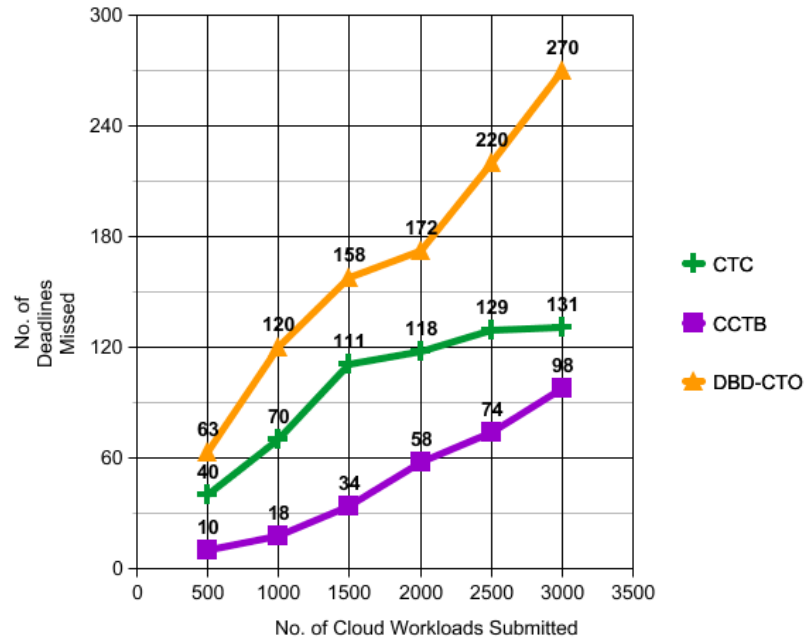


Figure 6.37: Number of Missed Deadlines

Test Case 8: Execution Time Variation

The execution time is decreasing with the increase budget as shown in Figure 6.38. In this algorithm, minimum cost for execution is \$50. At a minimum cost, the execution time is larger. With the increase in budget, the more number of resources provided to reduce the execution time and number of deadlines missed are also decreasing.

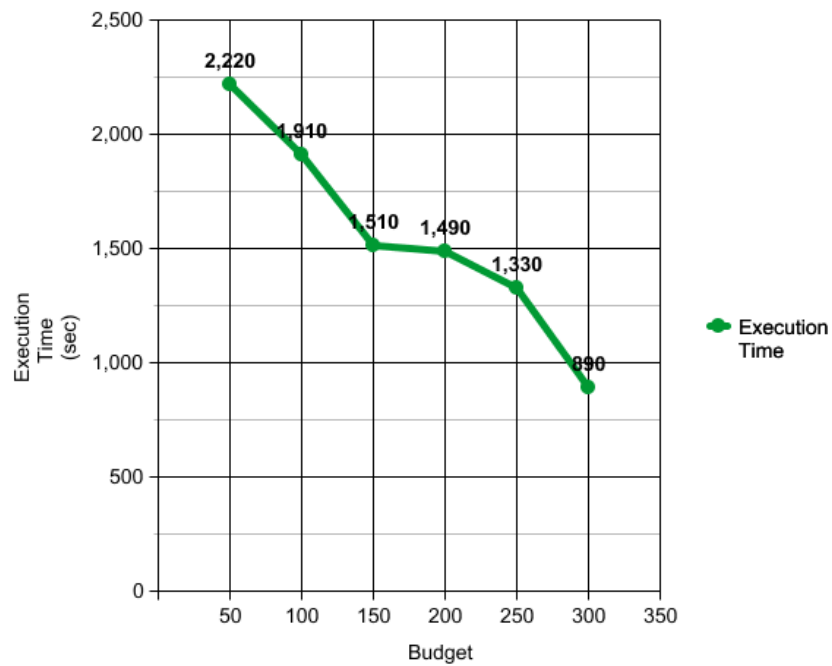


Figure 6.38: Execution Time Variation

Test Case 9: Execution Cost Variation

The cost of executing the Cloud workloads is varying with the increase in allocated budget as shown in Figure 6.39.

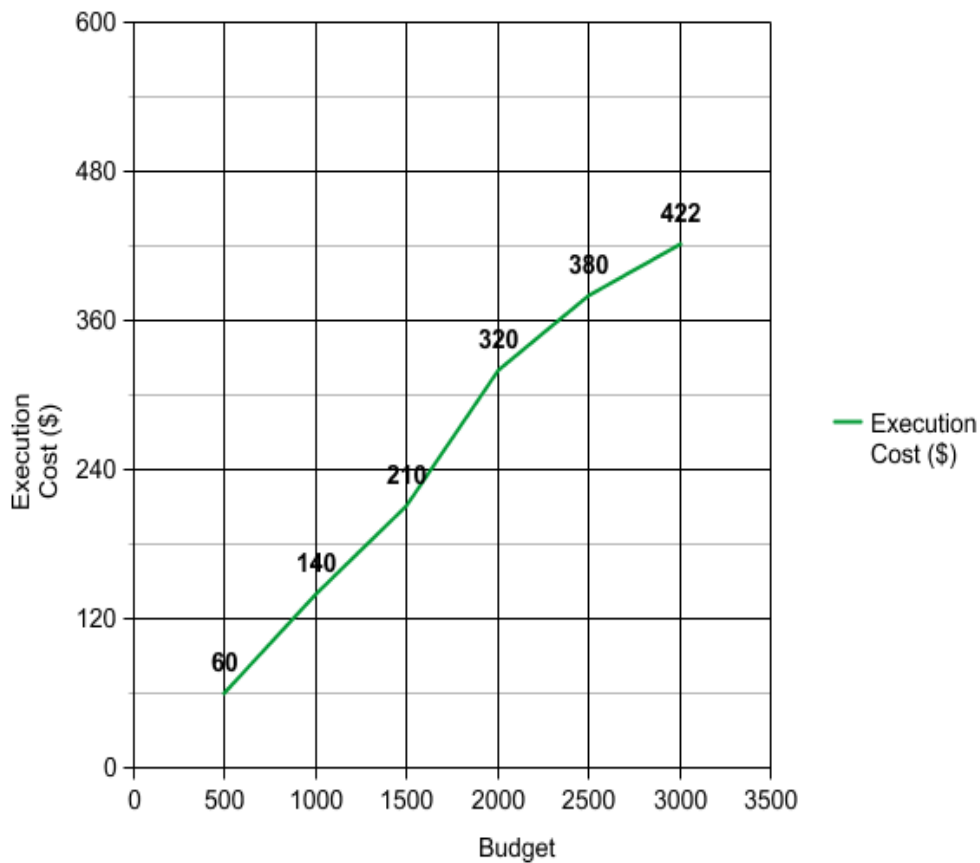


Figure 6.39: Execution Cost Variation

Test Case 10: Energy Consumption

For different number of resources there is fixed threshold value of energy consumed during Cloud workload execution. The calculated energy consumption is compared with threshold value and the Cloud workload is executed only if the calculated energy consumption is less than or equal to threshold value. For successful execution of a Cloud Workload, the Actual Energy Consumption (E_{cloud}) will be less than Threshold Energy Value ($E_{Threshold}$). Figure 6.40 shows the comparison of actual energy consumption and threshold energy value.

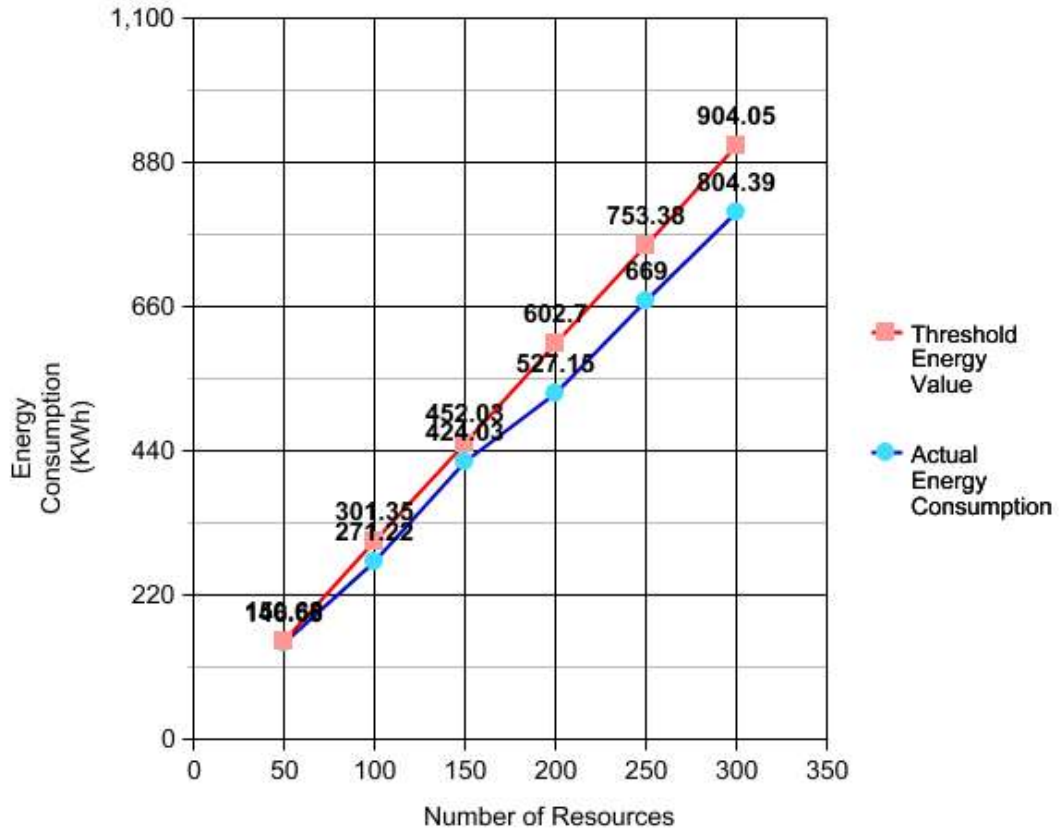


Figure 6.40: Energy Consumption and Number of Resources

6.5.3 Cost Based (CB) Scheduling Policy

Allocate Resources based on cost, the workload which has more budget (B_E) will execute first. If the two workloads have same budget then that workload will execute first that has lesser execution time. By default, PS=1.

B_E	265	252	200	170	155	120	100	72	65	62
workload	W8	W7	W6	W4	W5	W3	W1	W9	W10	W2
Resource	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10

The implementation of Cost Based Scheduling Policy follows the algorithm shown in Figure 5.5. First, the allocation agent begins to compute the cost of each Cloud workload then sorted as the priority given to the Cloud workload which has maximum budget. The allocation agent then schedules all the workloads with high budget request to the resources that provide high QoS. Finally, all other workloads are scheduled on the available resources set.

6.5.4 Time Based (TB) Scheduling Policy

Allocate Resources based on time, the workload which has shortest deadline time (D_t) will execute first. If the two workloads have same deadline time then that workload will execute first that has lesser execution time. By default, PS=1. In this scenario, there are three workloads (W2, W7, W9) have same deadline time, the sorted according to the maximum budget ($W7 > W9 > W2$).

D_t	2	4	4	4	6	10	12	14	20	21
workload	W6	W7	W9	W2	W3	W5	W1	W10	W8	W4
Resource	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10

Number of Deadlines Missed: 3 (W2, W9, W10)

To execute W2, W9 and W10 within their deadline, change budget according to the Processing Speed, and add cost \$50 if user want to double the Processing Speed (See Table 6.3). With PS=2, these three workloads will be execute before deadline, by recalculating the value of E_t as described in Table 6.8.

Table 6.8: Actual and Improved Execution Time

WId	D_t (H)	Actual E_t (PS=1)	Improved E_t (PS=2)
W2	4	6.45	3.22
W9	4	5.55	3.17
W10	14	21.53	12.17

Now the workloads W2, W9 and W10 will be execute before their deadline.

The implementation of Time Based Scheduling Policy follows the algorithm shown in Figure 5.6. First, the allocation agent begins to compute the Deadline Time of the Cloud workload on the given budget. The allocation agent then schedules all the Cloud workloads with smallest execution time request to the resources that provide high QoS. If any deadline found missed then recalculate the execution time by increasing the value of processing speed (PS) and it will increase cost only.

Test Case 1: Completion Time vs. Allocated Budget

The Cloud workload is being executed with different constraints. There is a fixed deadline i.e. 1500 seconds; the Cloud workload should be executed successfully

before deadline. In this experiment, different budget was allocated. In First Come First Serve (FCFS) Based Scheduling Policy, the execution time is larger at different budget. The execution time in Cost Based Scheduling Policy is lesser than FCFS but more than Time Based Scheduling Policy. The execution time is lesser in Time Based Scheduling Policy at maximum budget that has been allocated. The execution time in both the Cost and Time Based Scheduling Policy is decreasing but remains same in FCFS as shown in Figure 6.41.

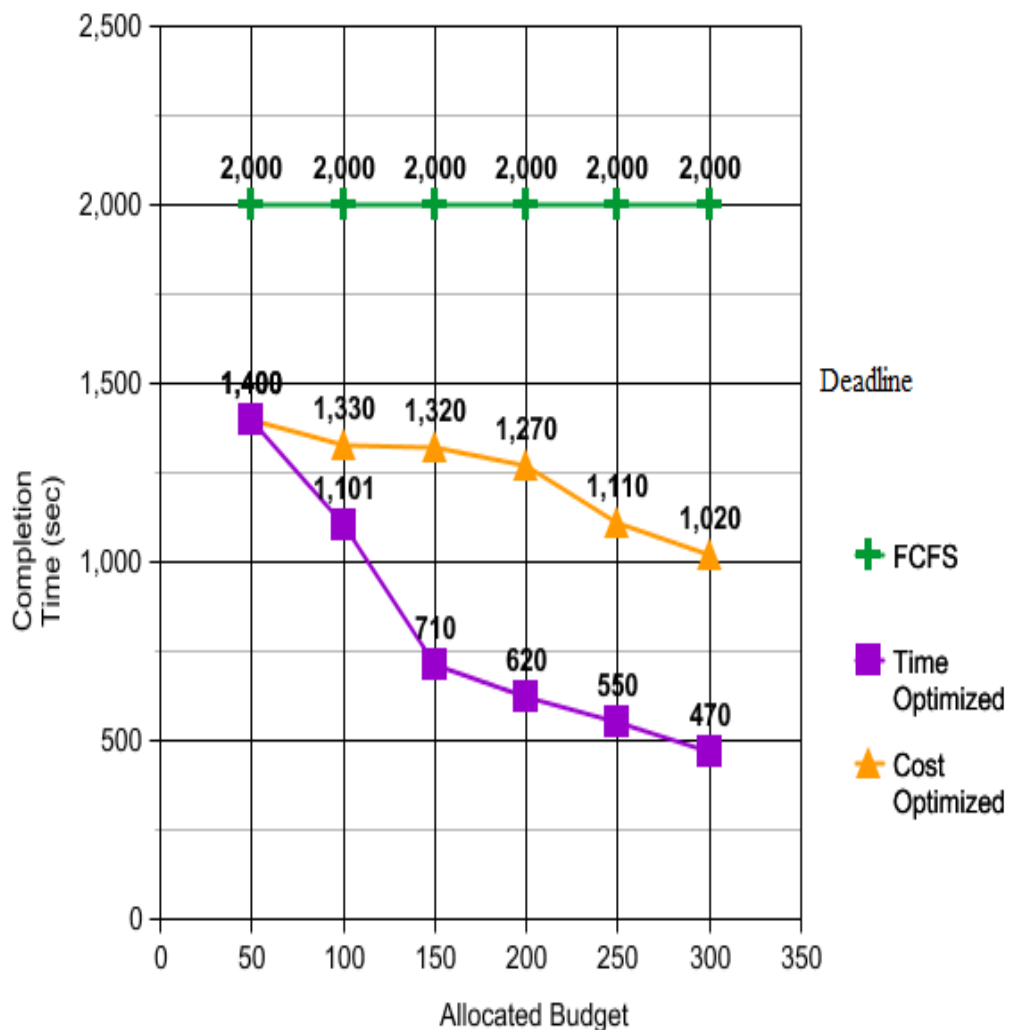


Figure 6.41: Completion Time vs. Allocated Budget

Test Case 2: Completion Time of Different Workloads

The execution time required to complete the Cloud workload is maximum in FCFS and minimum in Cost Based Scheduling Policy, but least in Time Based Scheduling Policy as shown in Figure 6.42. Both Cost and Time Based Scheduling Policy executes the Cloud workload before desired deadline.

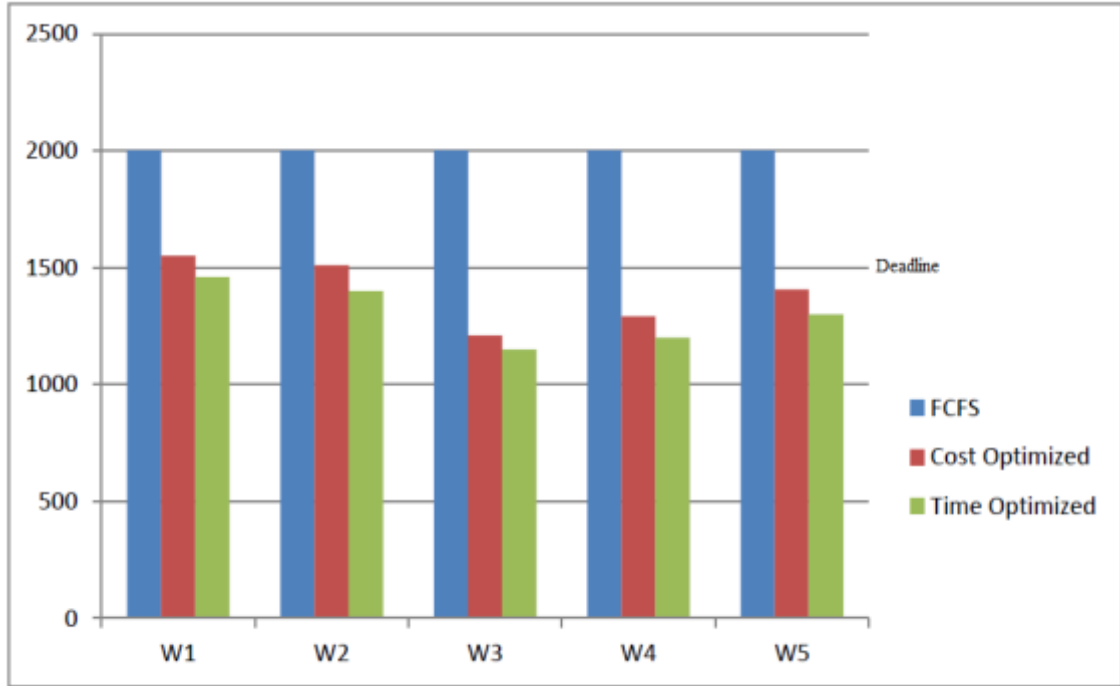


Figure 6.42: Completion Time of Different Workloads

6.5.5 Bargaining Based (BB) Scheduling Policy

Table 6.9 shows different Cloud workloads along with their parameters such as deadline, estimated budget and execution time.

Table 6.9: Cloud Workloads Detail

WId	W_d (H)	B_E (\$)	E_t
W1	12:00	100	12
W2	4:00	60	6.7
W3	6:00	120	5
W4	21:00	170	12.35
W5	10:00	150	6.7
W6	2:00	200	1
W7	4:00	250	1.6
W8	20:00	260	7.69
W9	4:00	70	5.71
W10	14:00	65	21.53

The various resources are available for the execution of above Cloud workloads along with their execution cost (C) and classified according to time slots (Hours) is described in Table 6.10.

Table 6.10: Resource Detail

Time Slot (H)									
0-2		2-4		4-8		8-16		16-32	
R	C (\$)	R	C (\$)	R	C (\$)	R	C (\$)	R	C (\$)
R1	100	R5	80	R8	105	R12	80	R16	160
R2	90	R6	75	R9	115	R13	85	R17	180
R3	110	R7	110	R10	125	R14	90	-	-
R4	120	-	-	R11	90	R15	70	-	-

The implementation complies with the negotiation among the various resources and Cloud workload producer along with different time slots. The allocation agent allocates the resources based on the bargaining between them as shown below:

E_t	12	6.7	5	12.35	6.7	1	1.6	7.69	5.71	21.53
Workload	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10
Resource	R15	R11	R11	R15	R11	R2	R2	R11	R11	R16

The implementation of Bargaining Based Scheduling Policy (BBSP) follows the algorithm shown in Figure 5.7. The implementation complies with the negotiation among the various resources and Cloud workload producer along with different time slots. The allocation agent allocates the resources based on the bargaining between them. The number of missed deadlines in both the cases is shown in Figure 6.43.

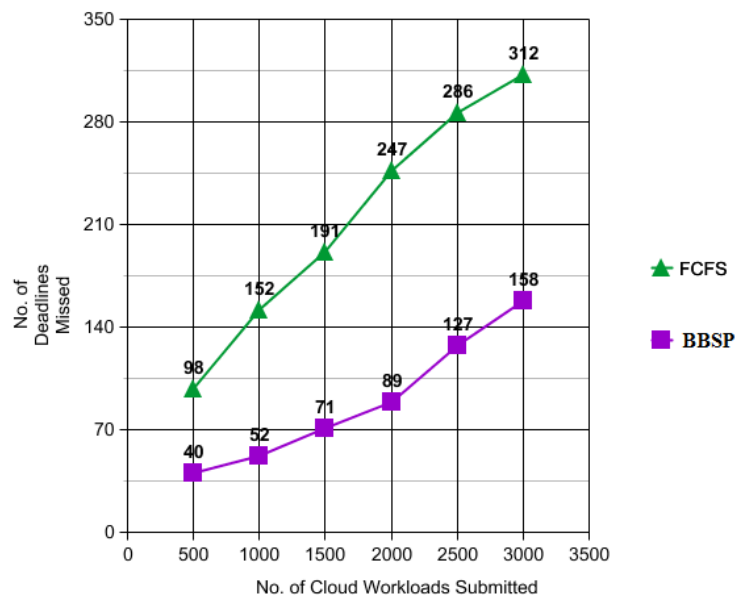


Figure 6.43: Number of Deadlines Missed

With the increase in number of Cloud workloads, the rate of missed deadlines is increased. In both the policies, the number of deadlines missed is varying. The lesser number of deadlines missed in BBSP as compared to FCFS.

6.6 Conclusion

This chapter presented the implementation of proposed framework and comparison of proposed framework with different traditional scheduling algorithms. The policy in which Cloud provider minimizing the cost as well as execution time along with least energy consumption is called CCTB Scheduling Policy. CCTB reduces the execution time by up to 30.94% compared to DBD-CTO and 6.54% for CTC. CCTB reduces the execution cost by up to 50% compared to DBD-CTO and 22.72% for CTC. The results depict that by increasing the number of resources, the execution time decreases. The costs of resources are decreasing with increasing the number of resources. The CCTB algorithm executes the same number of Cloud workloads at a minimum cost. The execution time is increasing with the increase in number of Cloud workloads and the execution time of CCTB for same number of Cloud workloads is slightly lesser than CTC. The existing algorithm based workload's execution resulted in a schedule which is expensive in comparison to the CCTB algorithm. From all the experimental results, the workload execution using the CCTB algorithm performs better. The overall cost for Cloud consumer's workload execution is less. The number of deadlines missed in Deadline and Budget Distribution - Based Cost Time Optimization (DBD-CTO) are maximum and minimum in Compromised Cost Time Based (CCTB). With the increase in budget, the more number of resources provided to reduce the execution time and number of deadlines missed are also decreasing. The cost of executing the Cloud workloads is varying with the increase in allocated budget. The Actual Energy Consumption (E_{cloud}) reduces up to 9.99% at 100 resources and 11.02% at 300 resources. Next chapter will discuss the conclusion and future work.

Conclusions and Future Scope

This chapter summarizes the research work presented in this thesis and highlights the main contributions. It also discusses open research problems in the area and outlines a number of future research directions.

7.1 Conclusions

Cloud Computing and its vital characteristics have been discussed in this thesis. This thesis focuses on the resource scheduling challenges that Cloud Computing is facing today. Several resource scheduling algorithms are compared with respect to the Cloud workload as an answer for the dynamic scalability of resources. This thesis also gives an insight about the problem of making decisions based on cost and time constraints in Cloud Computing. The different Cloud workloads and design patterns have been identified and analyzed. The key QoS requirements for every Cloud workload have been identified. The clustering of these Cloud workloads is done through K-Means Clustering Algorithm by assigning the appropriate weights to the different quality attributes. The results of clustering of Cloud workloads have been presented with the help of WEKA tool. Further, workload based Cloud Framework is proposed and implemented in this work. The experimental results gathered through CloudSim 3.0 clearly demonstrate that the proposed framework has better performance in terms of execution time, cost and energy consumption as compared to existing scheduling algorithms.

7.2 Thesis Contribution

- a) In this thesis, currently available resource scheduling algorithms like SHEFT, OWS, RASA, ACO, PSO etc. are discussed and compared.
- b) Currently available Resource Scheduling Policies like processing time based, VM based, policy based, hardware resource dependency based, bidding based, application based, utility based and SLA based are discussed.

- c) The different Cloud workloads and design patterns have been identified, analyzed and mapped.
- d) Both homogenous and heterogeneous workload based framework is proposed and designed.
- e) The clustering of Cloud workloads is done through K-Means Clustering Algorithm and results are presented using WEKA tool.
- f) Four resource scheduling policies (Compromised Cost - Time Based (CCTB) Scheduling Policy, Time Based (TB) Scheduling Policy, Cost Based (CB) Scheduling Policy and Bargaining Based (BB) Scheduling Policy) are proposed.
- g) Decision tree based scheduling criteria is discussed.
- h) Cloud Workload Management Portal is designed and implemented and proposed framework is validated with CloudSim 3.0.

7.3 Future Scope

- a) The proposed framework presented in this thesis can be extended further to add different Cloud providers.
- b) IaaS providers can use these results to quickly assess possible reductions in execution time and cost, hence having the potential to save energy.
- c) To conduct further evaluations of proposed framework with realistic test bed.

References

- [1] A. Michael, F. Armando , . G. Rean , D. J. Anthony , K. Randy , . K. Andy , L. Gunho , P. David , R. Ariel , S. Ion and Z. Matei , "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50-58, 2010.
- [2] "Vision of key characteristics of cloud environment," corelynx, [Online]. Available: <http://www.corelynx.com/services/cloud-computing.html>. [Accessed 15 1 2013].
- [3] J. Cloud, "The Sharing of resources among Cloud Consumers," Computer History Museum, [Online]. Available: <http://www.computerhistory.org/press/babbage-engine-exhibit.html>. [Accessed 29 1 2013].
- [4] Q. Zhang, L. Cheng and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *Journal of Internet Services and Applications*, vol. 1, no. 1, pp. 7-18, 2010.
- [5] R. Buyya, C. S. Yea, S. Venugopala, J. Broberga and I. Brandicc, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599-616, 2009.
- [6] P. Mell and T. Grance, "The NIST definition of cloud computing (draft)," *The NIST Definition of Cloud Computing, NIST Special Publication 800-145*, 2011.
- [7] C. Moroney, R. Davies and J.-P. Muller, "Operational retrieval of cloud-top heights using MISR data," *IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING*, vol. 40, no. 7, pp. 1532-1540, 2002.
- [8] L.-J. Zhang and Q. Zhou, "CCOA: Cloud computing open architecture," in *IEEE International Conference on Web Services*, Los Angeles, 2009.
- [9] S. Verbrugge, D. Colle, M. Pickavet, P. Demeester, S. Pasqualini, A. Iselt, A. Kirstädter, R. Hülsermann, F.-J. Westphal and M. Jäger, "Methodology and input availability parameters for calculating OpEx and CapEx costs for realistic network scenarios," *Journal of Optical Networking*, vol. 5, no. 6, pp. 509-520, 2006.
- [10] Z. Shen, S. Subbiah, X. Gu and J. Wilkes, "Cloudscale: elastic resource scaling for multi-tenant cloud systems," in *SOCC '11 Proceedings of the 2nd ACM*

Symposium on Cloud Computing, Cascais, Portugal, 2011.

- [11] "The seven elements of Cloud Computing Value," MWD, [Online]. Available: <http://www.mwdadvisors.com/blog/2009/06/seven-elements-of-cloud-computings.html>. [Accessed 10 2 2013].
- [12] M. Rui, "The Exploration of Library Automation Management System in Cloud Computing Environment, Researches in Library Science," *Researches in Library Science*, 2009. [Online]. Available: http://en.cnki.com.cn/Article_en/CJFDTOTAL-TSSS200907011.htm. [Accessed 24 1 2013].
- [13] B. P. Rimal and E. Choi, "A taxonomy and survey of cloud computing systems," in *Fifth International Joint Conference on INC, IMS and IDC*, Seoul, Korea, 2009.
- [14] O. Brian, T. Brunschwiler, H. Dill, H. Christ, B. Falsafi, M. Fischer and S. G. Grivas, "Cloud Computing," *Communications of the ACM*, vol. 51, no. 7, pp. 9-11, 2008.
- [15] T. Anderson, D. Culler and D. Patterson , "A case for networks of workstations (NOW)," in *Symposium Record of Hot Interconnects II*, pp. 24-39, 1994.
- [16] I. Foster, Y. Zhao, I. Raicu and S. Lu , "Cloud computing and grid computing 360-degree compared," in *In Grid Computing Environments Workshop GCE '08*, 2008.
- [17] R. Gruber, P. Volgersb, A. . D. Vitac, M. Stengelc and T.-M. Trand, "Parameterisation to tailor commodity clusters to applications," *Future Generation Computer Systems*, vol. 19, no. 1, pp. 111-120, 2003.
- [18] O. Ingthorsson, "Complete Cloud Computing Resource," *Cloud Computing Topics*, [Online]. Available: <http://cloudcomputingtopics.com/2013/01/cloud-computing-what-are-iaas-paas-saas/>. [Accessed 18 1 2013].
- [19] B. P. Rimal, A. Jukan, D. Katsaros and Y. Goeleven, "Architectural requirements for cloud computing systems: an enterprise cloud approach," *Journal of Grid Computing*, vol. 9, no. 1, pp. 3-26, 2011.
- [20] J. D. Jesús, "Cloud deployment models," IBM, [Online]. Available: http://www.ibm.com/developerworks/websphere/techjournal/1206_dejesus/1206_dejesus.html. [Accessed 14 1 2013].
- [21] T. Dillon , C. Wu and E. Chang, "Cloud Computing: Issues and Challenges," in

24th IEEE International Conference on Advanced Information Networking and Applications, Perth, Australia, 2010.

- [22] V. K. Reddy, B. T. Rao and L. S. Re, "Research issues in Cloud Computing," *Global Journal of Computer Science and Technology*, vol. 11, no. 11, 2011.
- [23] B. Li, J. Li, J. Huai, T. Wo, Q. Li and L. Zhong, "EnaCloud: an energy-saving application live placement approach for loud computing environments," in *IEEE International Conference on Cloud Computing*, Bangalore, India, 2009.
- [24] W. Crine and F. Berman, "A comprehensive model of the supercomputer workload," in *Fourth Annual IEEE International Workshop on Workload Characterization. WWC-4*, Austin, Texas, 2001.
- [25] M. . F. Arlitt and C. . L. Williamson, "Web server workload characterization: The search for invariants," in *ACM SIGMETRICS international conference on Measurement and modeling of computer systems, SIGMETRICS '96*, Philadelphia, PA, USA, 1996.
- [26] L. Cherkasova and M. Gupta, "Characterizing locality, evolution, and life span of accesses in enterprise media server workloads," in *12th international workshop on Network and operating systems support for digital audio and video*, FL, USA, 2002.
- [27] D. Gmach, J. Rolia , L. Cherkasova and A. Kemper, "Workload analysis and demand prediction of enterprise data center applications," in *IEEE 10th International Symposium on Workload Characterization, IISWC '07*, Boston, MA, USA, 2007.
- [28] N. Bobroff, A. Kochut and K. Beaty, "Dynamic placement of virtual machines for managing SLA violations," in *IFIP/IEEE Integrated Network Management*, Bombay, 2007.
- [29] A. Verma , G. Dasgupta , T. Kumar and N. Prad, "Server workload analysis for power minimization using consolidation," in *USENIX Annual technical conference*, San Jose, CA, 2009.
- [30] J. Rolia, . L. Cherkasova, . M. Arlitt and A. Andrzejak, "A capacity management service for resource pools," in *5th international workshop on Software and performance*, Illes Balears, Spain, 2005.

- [31] A. Khan, X. Yan, S. Tao and . N. Anerousis, "Workload characterization and prediction in the cloud: A multiple time series approach," in *Network Operations and Management Symposium (NOMS), IEEE*, Krakow, Poland, 2012.
- [32] A. Krogh N, B. Larsson, G. V. Heijne and E. L. L. Sonnhammer, "Predicting transmembrane protein topology with a hidden Markov model: application to complete genomes," *Journal of molecular biology*, vol. 305, no. 3, pp. 567-580, 2001.
- [33] S. J. Chen, P. H. Liang and J.-M. Yang, "Workload Evaluation and Analysis on Virtual Systems," in *IEEE International Conference on E-Business Engineering*, 2010, 2010.
- [34] R. V. d. Bossche, K. Vanmechelen and J. Broeckhove, "Cost-optimal scheduling in hybrid iaas clouds for deadline constrained workloads," in *IEEE 3rd International Conference on Cloud Computing*, Florida, USA, 2010.
- [35] P. Xiong , Z. Wang, S. Malkowski , Q. Wang , D. Jayasinghe and C. Pu , "Economical and robust provisioning of n-tier cloud workloads: A multi-level control approach.," in *Distributed Computing Systems (ICDCS)*, Minneapolis, Minnesota, 2011.
- [36] K. Tsakalozos, . M. Roussopoulos, V. Floros and A. Delis, "Nefeli: Hint-based execution of workloads in clouds," in *International Conference on Distributed Computing Systems*, Genova, Italy, 2010.
- [37] R. V. d. Bossche, K. Vanmechelen and J. Broeckhove, "Online cost-efficient scheduling of deadline-constrained workloads on hybrid clouds," *Future Generation Computer Systems*, vol. 29, no. 4, pp. 973-985, 2013.
- [38] G. Kousiouris, . T. Cucinotta and . T. Varvarigou, "The effects of scheduling, workload type and consolidation scenarios on virtual machine performance and their prediction through optimized artificial neural networks," *Journal of Systems and Software*, vol. 84, no. 8, pp. 1270-1291, 2011.
- [39] S. Mahambre, P. Kulkarni, U. Bellur, G. Chafle and D. Deshpande, "Workload Characterization for Capacity Planning and Performance Management in IaaS Cloud," in *IEEE Cloud Computing in Emerging Markets (CCEM)*, BANGALORE, INDIA, 2012.
- [40] A. Nahir, A. Orda and D. Raz, "Workload factoring with the cloud: A game-theoretic perspective," in *The 31st Annual IEEE International Conference on*

Computer Communications: Mini-Conference, FL, USA, 2012.

- [41] S. Pandey, L. Wu, S. Guru and R. Buyya, "A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments," in *Advanced Information Networking and Applications (AINA), 24th IEEE International Conference*, Perth, Australia, 2010.
- [42] M. Xu, L. Cui, H. Wang and . Y. Bi , "A multiple QoS constrained scheduling strategy of multiple workflows for cloud computing.," in *IEEE International Symposium on Parallel and Distributed Processing with Applications*, MA, USA, 2009.
- [43] H. Topcuoglu, S. Hariri and M.-Y. Wu, "Task scheduling algorithms for heterogeneous processors," in *Heterogeneous Computing Workshop, (HCW'99)*, San Juan, Puerto Rico, 1999.
- [44] Z. Wu , X. Liu, Z. Ni, D. Yuan and Y. Yang, "A market-oriented hierarchical scheduling strategy in cloud workflow systems," *The Journal of Supercomputing*, vol. 63, no. 1, pp. 256-293, 2013.
- [45] J. Yu, R. Buyya and C. K. Tham, "Cost-based scheduling of scientific workflow applications on utility grids," in *e-Science and Grid Computing, IEEE*, IL, USA, 2005.
- [46] R. Sakellariou, H. Zhao, E. Tsiakkouri and M. . D. Dikaiakos, "Scheduling workflows with budget constraints," *Integrated Research in GRID Computing*, pp. 189-202, 2007.
- [47] S. Parsa and . R. Entezari-Maleki, "RASA: A new task scheduling algorithm in grid environment," *World Applied Sciences Journal*, vol. 7, no. Special, pp. 152-160, 2009.
- [48] A. G. Delavar, M. Javanmard, M. B. Shabestari and M. . K. Talebi, "RSDC (Reliable Scheduling Distributed In Cloud Computing)," *International Journal of Computer Science, Engineering and Applications (IJCSEA)*, vol. 2, no. 3, pp. 1-16, 2012.
- [49] M. Dakshayini and H. S. Guruprasad, "An Optimal Model for Priority based Service Scheduling Policy for Cloud Computing Environment," *International Journal of Computer Applications*, vol. 32, no. 9, p. 0975 – 8887, 2011.

- [50] S. Ghanbari and . M. Othman, "A Priority based Job Scheduling Algorithm in Cloud Computing," *Procedia Engineering, International Conference on Advances Science and Contemporary Engineering*, vol. 50, pp. 778-785, 2012.
- [51] E.-S. T. El-kenawy, A. I. El-Desoky and M. F. Al-rahamawy, "Extended Max-Min Scheduling Using Petri Net and Load Balancing," *International Journal of Soft Computing and Engineering (IJSCE)*, vol. 2, no. 4, pp. 198-203, 2012.
- [52] S. Ambike, D. Bhansali, J. Kshirsagar and J. Bansiwai, "An Optimistic Differentiated Job Scheduling System for Cloud Computing," *International Journal of Engineering Research and Applications (IJERA)* , vol. 2, no. 2, pp. 1212-1214, 2012.
- [53] S. Selvarani and G. S. Sadhasivam, "Improved cost-based algorithm for task scheduling in cloud computing," in *IEEE, Computational Intelligence and Computing Research (ICCIC)*, Tamilnadu, INDIA, 2010.
- [54] I. A. Moschakis and H. D. Karatza, "Evaluation of gang scheduling performance and cost in a cloud computing system," *The Journal of Supercomputing*, vol. 59, no. 2, pp. 975-992, 2012.
- [55] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, IEEE*, 1995.
- [56] Z.-H. Zhan, J. Zhang, Y. Li and H. . S.-H. Chung, "Adaptive particle swarm optimization," *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS*, vol. 39, no. 6, pp. 1362-1381, 2009.
- [57] C. Shi-weia and P. Yub, "Credibility-based Dynamic Resource Distribution Strategy Under Cloud Computing Environment," *Computer Engineering*, vol. 11, 2011.
- [58] G. V. Valkenhoef, S. D. Ramchurn, P. Vytelingum, N. R. Jennings and R. Verbrugge, "Continuous Double Auctions with Execution Uncertainty," *Agent-Mediated Electronic Commerce. Designing Trading Strategies and Mechanisms for Electronic Markets, Lecture Notes in Business Information Processing*, vol. 59, pp. 226-241, 2010.
- [59] H. Xia-yu, Z. Jun and H. Wen-xin, "Ant colony optimization algorithm for computing resource allocation based on cloud computing environment," *Journal of East China Normal University (Natural Science)*, pp. 127-134, 2010.

- [60] B. Schroede and G. A. Gibson, "A large-scale study of failures in high-performance computing systems," in *DSN '06 Proceedings of the International Conference on Dependable Systems and Networks*, 2006.
- [61] R. K. Sahoo, A. Sivasubramaniam, M. S. Squillante and Y. Zhang, "Failure data analysis of a large-scale heterogeneous server environment," in *International Conference on Dependable Systems and Networks*, 2004.
- [62] M. Dorigo, V. Maniezzo and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions*, vol. 26, no. 1, pp. 1-13, 1996.
- [63] W. Yonggui and H. Ruilian, "Study on cloud computing task schedule strategy based on MACO algorithm," *Computer Measurement & Control*, vol. 19, no. 5, pp. 1203-1204, 2011.
- [64] W. Jiyi, P. Lingdi, P. Xuezheng and L. Zhuo, "Cloud Computing: Concept and Platform," *Telecommunications Science*, vol. 12, pp. 23-30, 2009.
- [65] L. Jian-feng and P. Jian, "Task scheduling algorithm based on improved genetic algorithm in cloud computing environment," *Journal of Computer Applications*, vol. 31, no. 1, pp. 184-186, 2011.
- [66] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *USENIX Association OSDI '04: 6th Symposium on Operating Systems Design and Implementation*, vol. 51, no. 1, pp. 107-113, 2008.
- [67] J. Li, M. Qiu, J.-W. Niu, Y. Chen and Z. Ming, "Adaptive resource allocation for preemptable jobs in cloud systems," in *Intelligent Systems Design and Applications (ISDA), IEEE*, 2010.
- [68] D. Shin and H. Akkan, "Domain-based virtualized resource management in cloud computing," in *Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, 2010.
- [69] K. Kumar, J. Feng, Y. Nimmagadda and Y.-H. Lu, "Resource Allocation for Real-Time Tasks using Cloud Computing," in *Computer Communications and Networks (ICCCN)*, 2011.
- [70] D. Gmach, J. Rolia and L. Cherkasova, "Satisfying service level objectives in a self-managing resource pool," in *Self-Adaptive and Self-Organizing Systems*,

IEEE, SASO'09, 2009.

- [71] X. Zhu, D. Young, B. J. Watson, Z. Wang, J. Rolia, S. Singhal , B. McKee, C. Hyser, D. Gmach, R. Gardner, T. Christian and L. Cherkasova, "1000 islands: Integrated capacity and workload management for the next generation data center," in *International Conference on Autonomic Computing, IEEE*, 2008.
- [72] D. Minarolli and . B. Freisleben, "Utility-based resource allocation for virtual machines in Cloud computing," in *IEEE Symposium on Computers and Communications (ISCC)*, 2011.
- [73] W. Hu, C. Tian, X. Liu, H. Qi, L. Zha , H. Liao, Y. Zhang and J. Zhang , "Multiple-Job Optimization in MapReduce for Heterogeneous Workloads," in *Sixth IEEE International Conference on Semantics, Knowledge and Grids*, 2010.
- [74] W.-Y. Lin, G.-Y. Lin and H.-Y. Wei, "Dynamic auction mechanism for cloud resource allocation," in *10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, 2010.
- [75] T. T. Huu and J. Montagnat, "Virtual resources allocation for workflow-based applications distribution on a cloud infrastructure," in *10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, 2010.
- [76] Y. C. Lee, C. Wang, A. Y. Zomaya and B. . B. Zhou, "Profit-driven Service Request Scheduling in Clouds," in *10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, 2010.
- [77] L. Wu, S. K. Garg and R. Buyya, "SLA –based Resource Allocation for SaaS Providers in Cloud Computing Environments," in *11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, 2011.
- [78] Y. Yang, K. Liu and J. Chen, "An algorithm in SwinDeW-C for scheduling transaction-intensive cost-constrained cloud workflows," in *Fourth IEEE International Conference on eScience*, 2008.
- [79] K. Liu, H. Jin, J. Chen, X. Liu, . D. Yuan and Y. Yang, "A compromised-time-cost scheduling algorithm in swindow-c for instance-intensive cost-constrained workflows on a cloud computing platform," *International Journal of High Performance Computing Applications*, vol. 24, no. 4, pp. 445-456, 2010.
- [80] P. Varalakshmi, A. Ramaswamy and A. Balasub, "An optimal workflow based scheduling and resource allocation in cloud," *Advances in Computing and*

Communications, vol. 190, pp. 411-420, 2011.

- [81] C. Lin, S. Lu, A. Balasubramanian and P. Vijaykumar, "Scheduling scientific workflows elastically for cloud computing,," in *IEEE International Conference on Cloud Computing (CLOUD)*, 2011.
- [82] J. Vlissides, R. Helm, R. Johnson and E. Gamma, *Design Patterns: Elements of Reusable Object-oriented Software*, Addison-Wesley, 1995.
- [83] C. Fehling, F. Leymann, R. Retter, D. Schumm and W. Schupeck, "An Architectural Pattern Language of Cloud-based Applications," in *Conference on Pattern Languages of Programs (PLoP)*, 2011.
- [84] C. Fehling, F. Leymann, R. Mietzner and W. Schupeck, "A Collection of Patterns for Cloud Types, Cloud Service Models, and Cloud-based Application Architectures," Report No. 2011/05, University of Stuttgart, 2011.
- [85] U. Zdun and P. Avgeriou, "Modeling Architectural Patterns Using Architectural Primitives," ACM SIGPLAN conference on Object oriented programming systems languages and applications, 2005.
- [86] G. Meszaros and J. Doble, *A Pattern Language for Pattern Writing*, Pattern languages of program design, Addison-Wesley, 1998.
- [87] G. Hohpe and B. Wolf, *Enterprise Integration Patterns: Designing, Building, and Deploying*, Addison-Wesley, 2004.
- [88] M. Fowler, *Patterns of Enterprise Application Architecture*, Addison-Wesley, 2003.
- [89] C. Fehling, F. Leymann, J. Rütshlin and D. Schumm, "Pattern-based Development and Management of Cloud Applications," *Future Internet*, vol. 4, pp. 110-141, 2012.
- [90] K. Hashizume, N. Yoshioka and E.B. Fernandez, "Misuse Patterns for Cloud Computing," in *Asian Conference on Pattern Languages of Programs (AsianPLoP)*, 2011.
- [91] A. Nowak, F. Leymann, D. Schleicher, D. Schumm and S. Wagner, "Green Business Process Patterns," in *Proceedings of the Conference on Pattern Languages of Programs (PLoP)*, 2011.

- [92] S. Riley, "How to Think Cloud Architectural Design Patterns for Cloud Computing," Cenrtal Ohio Agile Association, [Online]. Available: <http://www.cohaa.org/content/content/how-think-cloud-architectural-design-patterns-cloud-computing>. [Accessed 28 12 2012].
- [93] J. R. Quinlan, C4.5: Programs for machine learning, Morgan Kaufmann , 1993.
- [94] M. Umanol, H. Okamoto, I. H. H. I. R. O. Y. U. K. I. Tamura, F. Kawachi, S. Umedzu and a. J. Kinoshita, "Fuzzy decision trees by fuzzy ID3 algorithm and its application to diagnosis systems," in *Fuzzy Systems. IEEE World Congress on Computational Intelligence*, 1994.
- [95] J.R.Quinlan, "Decision trees and decision making," *IEEE transaction on system Man cyber*, vol. 20, no. 2, pp. 339-346, 1990.
- [96] R. Kohavi, "Scaling up the accuracy of naive-Bayes classifiers: A decision-tree hybrid," in *PROCEEDINGS OF THE SECOND INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING*, 1996.
- [97] M. Panda and M. R. Patra, "A comparative study of data mining algorithms for network intrusion detection," in *Emerging Trends in Engineering and Technology, ICETET '08.*, 20008.
- [98] N. L. son and P. Davidson, "Multi-dimensional measures function for classifier performance," in *2nd. IEEE International conference on Intelligent system*, 2004.
- [99] K. Omer, I. Maljevic, R. Anthony, . M. Petridis, K. Parrott and M. Schulz, "Dynamic scheduling of virtual machines running HPC workloads in scientific grids," in *3rd International IEEE Conference on New Technologies, Mobility and Security (NTMS)*, 2011.
- [100] "Cloud Workloads," CloudRoad, [Online]. Available: <http://www.1cloudroad.com/cloud-infrastructure-providers-for-2013>. [Accessed 11 2 2013].
- [101] W.-T. Tsai, X. Sun and J. Balasooriya, "Service-Oriented Cloud Computing Architecture," in *Seventh International Conference on Information Technology*, 2010.
- [102] G. Abowd, L. Bass, P. Clements, R. Kazman, L. Northrop and A. Zaremski, Recommended Best Industrial Practice for Software Architecture Evaluation,

Pennsylvania: Software Engineering Institute, Carnegie Mellon University, 1997.

- [103] E. J. Mishan, *Cost-benefit analysis: an informal introduction*, KY, U.S.A.: Routledge, Chapman & Hall, Incorporated, 1988.
- [104] K. Shvachko, H. Kuang, S. Radia and R. Chansler, "The Hadoop Distributed File System," in *IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, 2010.
- [105] B. W. Copeland and J. I. Shuval, "Computer-based uniform data interface (UDI) method and system using an application programming interface (API)". US Patent 5815703, 29 9 1998.
- [106] P. Hartman and O. Rutz, "Decimal Floating Point Computations in SAP NetWeaver 7.10," IBM White Paper, 2007.
- [107] R. . N. Calheiros, R. Ranjan and R. Buyya, "Virtual machine provisioning based on analytical performance and qos in cloud computing environments," in *International Conference on Parallel Processing, ICPP '11*, 2011.
- [108] T. Cucinotta, D. Giani, D. Faggioli and F. Checconi, "Providing Performance Guarantees to Virtual Machines Using Real-Time Scheduling," in *Euro-Par 2010 Parallel Processing Workshops, LNCS, Springer*, 2011.
- [109] M. A. El-Refaey and M. A. Rizkaa, "Virtual systems workload characterization: An overview.," in *18th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises, WETICE '09.*, 2009.
- [110] A. S. M. Kunz and R. R. Dumk, "Towards a service-oriented measurement infrastructure," in *3rd Software Measurement European Forum (SMEF)*, 2006.
- [111] L. Rosenberg, T. Hammer and J. Shaw, "Software metrics and reliability," in *9th International Symposium*, Germany , 1998.
- [112] L. H. Rosenberg and L. E. Hyatt, "Software quality metrics for object-oriented environments," SATC's research on OO metrics, 1997.
- [113] S. Zhao, X. Lu, X. Zhou, T. Zhang and J. Xue, "A Software Reliability Model For Web Services From the consumers' perspective," *International Conference on Computer Science and Service System (CSSS)*, 2011.
- [114] L. Shen and S. Ren, "Analysis and measurement of software flexibility based on

flexible point," Singapore National Employers' Federation (SNEF), Italy, 2006.

- [115] S. Suakanto, S. S. H. Supangkat and R. Saragih, "Performance Measurement of Cloud Computing Services," *International Journal on Cloud Computing: Services and Architecture (IJCCSA)*, vol. 2, no. 2, pp. 9-20, 2012.
- [116] L.-J. Zhang and J. Zhang , "Architecture-driven variation analysis for designing cloud applications," in *IEEE International Conference on Cloud Computing*, 2009.
- [117] L. Tang, J. Don, Y. Zhao and L. Zhang, "Enterprise Cloud Service Architecture," in *IEEE 3rd International Conference on Cloud Computing*, 2006.
- [118] V. Makhija, B. Herndon, P. Smith, L. Roderick, E. Zamost and J. Anderson, "VMmark: A Scalable Benchmark for Virtualized Systems," in *Technical Report VMware*, 2006.
- [119] M. R. J. Qureshi and W. A. Qureshi, "Evaluating Requirement Specification Document to Improve the Quality of Software," *AWERProcedia Information Technology and Computer Science*, vol. 1, pp. 596-600, 2012.
- [120] M. A. Elghany, N. Khalifa and M. A. Elghany, "Quantifying Software Reliability Attribute through the Adoption of Weighting Approach to Functional Requirements," in *International Conference on Software and Computer Applications (ICSCA 2012)*, Singapore, 2012.
- [121] M. Saeid, A. A. A. Ghani and H. Selamat, "Rank-Order Weighting of Web Attributes for Website Evaluation," *The International Arab Journal of Information Technology*, vol. 8, no. 1, pp. 30-38, 2011.
- [122] R. G. Dromey, "A model for software product quality," *IEEE Transactions on Software Engineering*, vol. 21, no. 2, pp. 146-462, 1995.
- [123] S. U. Malik, "Customer Satisfaction, Perceived Service Quality and Mediating Role of Perceived Value," *International Journal of Marketing Studies*, vol. 4, no. 1, pp. 68-76, 2012.
- [124] V. Nallur and R. Bahsoon, "Design of a market-based mechanism for quality attribute tradeoff of services in the cloud," in *ACM Symposium on Applied Computing*, 2010.
- [125] A. Stefani and M. Xenos, "E-commerce system quality assessment using a model based on ISO 9126 and Belief Networks," *Software Quality Control*, vol. 16, no.

1, pp. 107 - 129 , 2008.

- [126] M. Davoudi and F. S. Aliee, "A new AHP-based approach towards Enterprise Architecture quality attribute analysis," in *Research Challenges in Information Science, RCIS*, 2009.
- [127] J. Garofalakis , A. Stefani, V. Stefanis and M. Xenos, "Quality Attributes of Consumer-based M-commerce Systems, White Paper," University of Patras, 2008.
- [128] C. Otieno, W. Mwangi and . S. Kimani, "Framework to Assess Software Quality in ERP Systems," in *Scientific Conference Proceedings*, 2012.
- [129] P. Clements, R. Kazman and M. Klein, *Evaluating software architectures*, Addison Wesley, 2001.
- [130] A. Meiappane, V. P. Venkatesan, N. Roshini, S. Nivedha and R. Maheswar, "Evaluation of Software Architecture Quality Attribute for an Internet Banking System," *International Journal of Scientific & Engineering Research*, vol. 4, no. 4, pp. 1704-1708, 2013.
- [131] A. Stefani and M. . N. Xenos, "Meta-metric Evaluation of E-Commerce-related Metrics," *Electronic Notes in Theoretical Computer Science (ENTCS)*, vol. 233, pp. 59-72 , 2009.
- [132] P. K. Bhattacharjee, "Service Quality Measurement with Minimum Attributes (SERVQUAL-MA) Technique Upgrade by Human Resource Development," *International Journal of Innovation, Management and Technology*,, vol. 1, no. 3, pp. 322-327, 2010.
- [133] M. R. Barbacci, "Software Quality Attributes and Architecture Tradeoffs," Software Engineering Institute, Carnegie Mellon University, Pittsburgh, 2003.
- [134] P. Berander, L.-O. Damm, J. Eriksson, T. Gorschek, K. Henningsson, P. Jönsson, S. Kågström, D. Milicic, F. Mårtensson, K. Rönkkö and P. Tomaszewski , "Software quality attributes and trade-offs," Blekinge Institute of Technology , 2005.
- [135] J. Brooke, "SUS-A quick and dirty usability scale," Redhatch Consulting, UK, 1996.
- [136] R. . V. Singh and M. P. Bhatia, "Data clustering with modified K-means algorithm," in *IEEE-International Conference on Recent Trends in Information*

Technology, ICRTIT, Chennai, 2011.

- [137] G. K. Gupta, *Introduction to Data Mining with Case Studies*, New Delhi: PHI Learning Pvt. Ltd., 2006.
- [138] M.-C. Su and C.-H. Chou, "A Modified Version of the K-Means Algorithm with a Distance Based on Cluster Symmetry," *IEEE Transactions On Pattern Analysis And Machine Intelligence*, vol. 23, no. 6, pp. 674-680, 2001.
- [139] T. Kanungo, D. . M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman and A. Y. Wu, "An Efficient k-Means Clustering Algorithm: Analysis and Implementation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 881-892, 2002.
- [140] A. Mahendiran, N. Saravanan, N. V. Subramanian and N. Sairam, "Implementation of K-Means Clustering in Cloud Computing Environment," *Research Journal of Applied Sciences, Engineering and Technology*, vol. 4, no. 10, pp. 1391-1394, 2012.
- [141] J. Joseph, "Patterns For High Availability, Scalability and Computing Power," MSDN, 2009. [Online]. Available: <http://msdn.microsoft.com/en-us/magazine/dd727504.aspx>. [Accessed 12 11 2012].
- [142] A. Barseghyan, . M. Kupriyanov, I. Kholod, S. Yelizarov and M. Thess, *Analysis of Data and Processes: From Standard to Realtime Data Mining*, 2013.
- [143] C. V´azquez , E. Huedo, R. . S. Montero and I. M. Llorente, "Dynamic provision of computing resources from grid infrastructures and cloud providers," in *Workshops at the Grid and Pervasive Computing Conference*, 2009.
- [144] G. M. Llorente, R. Moreno-Vozmediano and R. S. Montero, "Cloud computing for on demand grid resource provisioning," in *Advances in Parallel Computing*, IOS Press, 2012, pp. 177 - 191.
- [145] M. A. Salehi and R. Buyya, "Adapting market-oriented scheduling policies for cloud computing," in *10th international conference on Algorithms and Architectures for Parallel Processing*, 2010.
- [146] A. P. Punnen, "The Traveling Salesman Problem: Applications, Formulations and Variations," in *The traveling salesman problem and its variations*, vol. 12, Springer, 2004, pp. 1-28.

- [147] "Microsoft Visual Studio 2010," Microsoft, [Online]. Available: <http://www.microsoft.com/visualstudio/eng/downloads>. [Accessed 2013 1 14].
- [148] "NetBeans IDE 7.1.2," NetBeans, [Online]. Available: <https://netbeans.org/community/news/show/1556.html>. [Accessed 14 1 2013].
- [149] R. . N. Calheiros, R. Ranjan, C. A. F. D. Rose and R. Buyya, "CloudSim: A novel framework for modeling and simulation of cloud computing infrastructures and services," Grid Computing and Distributed Systems Laboratory, The University of Melbourne, Australia, 2009.
- [150] A. Jana, "Java.NET : Integration of Java and .NET," Code Project, [Online]. Available: <http://www.codeproject.com/Articles/26144/Java-NET-Integration-of-Java-and-NET>. [Accessed 1 2 2013].
- [151] "SQL Server," Microsoft, [Online]. Available: <http://www.microsoft.com/en-us/sqlserver/get-sql-server/try-it.aspx> . [Accessed 12 2 2013].
- [152] M. Hall, E. Frank, G. Holmes , B. Pfahringer and I. . H. Witten, "The WEKA data mining software: an update," *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 1, pp. 10-18, 2009.
- [153] A. Verma and S. Kaushal, "Deadline and Budget Distribution based Cost-Time Optimization Workflow Scheduling Algorithm for Cloud," in *IJCA Proceedings on International Conference on Recent Advances and Future Trends in Information Technology (iRAFIT 2012)*, 2012.

List of Publications

Published

- [1] S. Singh and I. Chana, "Cloud Based Development Issues: A Methodical Analysis," *International Journal of Cloud Computing and Services Science (IJ-CLOSER)*, vol. 2, no. 1, pp. 73-84, 2012.

Communicated

- [2] S. Singh, I. Chana, " Green Computing: Energy based Efficient Resource Scheduling Framework for IaaS Cloud," in *The 16th IEEE International Conference on Computational Science and Engineering (CSE 2013), Sydney, Australia, 2013*.