A

Report On

Effective Resource Management in Clouds using Advance Reservation

PREPARED BY

Gayatri Sunilkumar
M.Tech(CSE)
08MCES54

GUIDED BY

Prof Madhuri Bhavsar

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INSTITUTE OF TECHNOLOGY
NIRMA UNIVERSITY
AHMEDABAD-382481

MAY - 2011

Effective Resource Management in Clouds using Advance Reservation

Major Project Part-II

Submitted in total fulfillment of the requirements

For the degree of

Master of Technology in Computer Science and Engineering

By

Gayatri. Sunilkumar

Roll No: 08MCES54

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
AHMEDABAD-382481

MAY, 2011

Declaration

This is to certify that

1. The thesis comprises my original work towards the degree of Master of Technology in Computer Science and Engineering at Nirma University and has not been submitted elsewhere for a degree or diploma.

2. Due acknowledgement has been made in the text to all other material used.

Gayatri Sunilkumar

CERTIFICATE

This is to certify that the Major Project Part-II entitled "Effective Resource Management in Clouds using Advance Reservation" submitted by Gayatri Sunilkumar (08MCES54), towards the total fulfillment of the requirements for the degree of Master of Technology in Computer Science and Engineering of Nirma University of Science and Technology, Ahmedabad is the record of work carried out by her under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for the examination. The contents embodied in this Major Project Part-II, to the best of my knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Prof. Madhuri.Bhavsar
Guide, Sr. Asso. Professor,
Department of Computer Science
and Engineering,
Institute of Technology,
Nirma University, Ahmedabad.

Dr. S.N.Pradhan
Professor and PG-Co-ordinator,
Department of Computer Science
and Engineering,
Institute of Technology,
Nirma University, Ahmedabad.

Prof.D.J.Patel
Professor and Head,
Department of Computer Science
and Engineering,
Institute of Technology,
Nirma University, Ahmedabad.

Dr. K.Kotecha
Director,
Institute of Technology,
Nirma University, Ahmedabad.

Effective Resource Management in Clouds using Advance Reservation

Gayatri Sunilkumar

Abstract

A Cloud is a type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers.Advance Reservation (AR) for Clouds is now a significant research focus as it allows consumers to gain synchronized access for their applications to be executed in parallel, and assures the availability of resources at specified future times. Clouds could be leased by the faculties of the college, who want to demonstrate the execution of the parallel programs on a distributed system during the lab hours or if a firm wants to lease the web server for months or years, a dedicated server in a data center may be from Amazon EC2 . The main objective of this work is to develop a Resource provisioning model for Eucalyptus Cloud. The Resource provisioning model proposed in this tshesis provides a support for best-effort provisioning and AR,suitable for the above specified scenarios. AR of Resources offers a better Quality of Service (QoS) for time critical applications. An effort is also made to discuss the possible states the Reservation made by a consumer may be in. This report also depicts how the ARM (Advance Resource Management) module can be embedded in the general architecture of the Eucalyptus Cloud. This thesis also confers a detailed discussion on Data Structures used for AR named as Eucalyptus Cloud Advance Reservation Queue (ECARQ) a terminology conied by me. Algorithms which would operate on the ECARQ are also discussed. Ubuntu Enterprise Cloud (UEC) which is the only Server with Eucalyptus cloud built in it, is used to test this Resource Provisioning Model.

Acknowledgements

It gives me immense pleasure in expressing my thanks and profound gratitude to **Prof. Madhuri Bhavsar,** Sr. Asso.Professor, Department of Computer Science and Engineering, Institute of Technology, Nirma University, Ahmedabad for her valuable guidance and continual encouragement throughout the Project. I'm heartily thankful to her for the timely suggestions and the clarity of the concepts of the topic that helped me a lot during this study.

I would like to extend my gratitude to **Dr.S.N.Pradhan,** Professor and M.Tech Coordinator, Department of Computer Science and Engineering, Institute of Technology, Nirma University, Ahmedabad for fruitful discussions and valuable suggestions during reviews and for the encouragement.

I would like to thank **Dr.Ketan Kotecha,** Hon'ble Director,Institute of Technology, Nirma University, Ahmedabad for providing basic infrastructure and healthy research environment.

Last, but not the least, no words are enough to acknowledge constant support and sacrifices of my family members **Mr. Sunilkumar my betterhalf** and **Aryan my son**, **My Mom Mrs. Kasturi Lengade and My Brother Mr. Rajendra Lengade** and his family because of whom I am able to complete my dissertation work successfully.

- Gayatri Sunilkumar
08MCES54

Contents

List of Figures

List of Algorithms

Abbreviation

AR Advance Reservation

ARM Advance Reservation Module

AMI Amazon Machine Image

CaaS Compiler as a Service

CLC Cloud Controller

CC Cluster Controller

CNs Compute Nodes

CPU Central Processing Unit

DbDaemon Database Daemon

EBS Elastic Block Storage

ECARQ Eucalyptus Cloud Advance Reservation Queue

Eucalyptus Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems

IaaS Infrastructure as a Service

IP Internet Protocol

JSP Java Server Pages

LAN Local Area Network

NC Node Controller

QoS Quality of Service

PaaS Platform as a Service

RMS Resource Management System

RS Reservation System

SLA Service Level Agreement

SaaS Software as a Service

SC Storage Controller

SEs Storage Elements

UEC Ubuntu Enterprise Cloud

VM Virtual Machine

Chapter 1

Introduction

This chapter presents a high level overview of this thesis. Today, the latest paradigm to emerge is that of Cloud computing which promises reliable services delivered through next-generation data centers that are built on compute and storage visualization technologies.Cloud computing is a new technology widely studied in recent years.

1.1 Cloud

A Cloud is a type of parallel and distributed system consisting of a collection of interconnected and visualized computers that are dynamically provisioned and presented as one or more unified computing resources based on Service-Level Agreements(SLA) established through negotiation between the service provider and consumers[1].

1.1.1 Popular available Clouds Platforms

- **Abicloud**
  Abicloud is a cloud computing platform that can be used to build, integrate and manage public as well as private cloud in the homogeneous environments . Using Abicloud, user can easily and automatically deploy and manage the server, storage system, network, virtual devices and applications [17].

- **Eucalyptus**
  Eucalyptus (Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems) [18] is an open-source implementation of Amazon EC2 and compatible with business interfaces. It also implements virtualization depending on Linux and Xen as EC2 does [17].

- **Nimbus**
  Nimbus is an open tool set, and also a cloud computing solution providing Infrastructure as a Service (IaaS) has supported many nonscientific research domain applications . It permit users lease remote resources and build the required computing environment through the deployment of virtual machines [17].

- OpenNebula
  OpenNebula is also an open source cloud service framework [2]. It allows user deploy and manage virtual machines on physical resources and it can set users data centers or clusters to flexible virtual infrastructure that can automatically adapt to the change of the service load [17].

1.2 Advance Reservation

Advance Reservation (AR) allows consumers to gain synchronized access for their applications to be executed in parallel, and assures the availability of resources at specified future times[7].AR is the process of requesting resources for use at specific times in the future . Common resources that can be reserved or requested are compute nodes (CNs), storage elements (SEs), network bandwidth or a combination of any of these.

1.3 Goal

- In this thesis an Advance Reservation Management (ARM) Module has been built for the Cloud Computing Platform especially Eucalyptus.
- AR in a Cloud allow Consumers to achieve synchronized access for their applications to be executed in parallel, and assures the availability of requested resources to the Consumers at specified future times. The AR Module would be aiming at the following points:
  - Balance the Load to obtain better Quality of Service (QoS).
  - Better Resource Utilization.
  - Reduce Fragmentation or idle time gaps.
  - Fewer Negotiations between the Consumer and the cloud AR system.
  - Better impact on income revenue ,reduced number of rejections and waiting time for local jobs.
- Develop a module useful to the Community of Cloud users.

1.4 Tools And Techniques

1.4.1 Ubuntu

Ubuntu 10.10 a.k.a Maverick Meerkat is the only server which comes with Ubuntu Enterprise Cloud (UEC) with Eucalyptus Cloud in-built [21]. It has many features discussed in the following section :

- Quality to the core : Ubuntu is part of the Debian family of Linux operating systems, which has the largest pool of developer talent, with every package built by experts. Canonical's rigorous release management, quality assurance, stress testing and product design enhances the quality of Ubuntu Server.

- **Modularity for large-scale deployment : Ubuntu's modularity means that the virtual appliances in the cloud are leaner, more efficient and more secure. The Ubuntu installation process is designed for people running thousands of similar, commodity servers, making it fast to deploy on a large scale and easy to maintain once deployed.**
- **Diversity of pre-packaged software : Ubuntu includes the largest selection of pre-packaged open-source software. In the cloud, these packages reduce the time to deploy new infrastructure and improve the reliability of the deployments.**
- **A renowned security track record :Ubuntu is protected by proactive security technology that defends the computer from typical threats. Canonical delivers updates for all supported software on Ubuntu, and publishes those updates globally. Ubuntu-based technology is easy to monitor for compliance with security policy. Ubuntu updates comply with industry-leading stable release update policy, ensuring that updates only address security and critical issues - avoiding unnecessary change.**
- **Certified on most of the hardware : Canonical certifies Ubuntu on a wide range of current hardware from leading manufacturers. They guarantee Ubuntu Enterprise Cloud on certified hardware that supports the new hardware visualization features in modern x86 servers (any recent x86 CPU).**
- **Tight integration between operating system and cloud infrastructure : Ubuntu leads the Linux field by integrating cloud capabilities directly into the operating system. UEC builds on the Eucalyptus cloud API standards. The relationship between Canonical and Eucalyptus Systems ensures that there is one clear path to resolve any issues with the operating system or cloud service.**
- **Compatible technology :Use the same Ubuntu machine images and management tools across both private and public systems, minimizing costly re-training or application change when moving from private to public and vice versa.**

1.4.2 Java

JDK version 1.6u20(jdk-6u20-linux-i586-rpm.bin) Geany is a light-weight IDE for Java on Ubuntu. It's in the Ubuntu repositories. As Eucalyptus is built in Java, an Object Oriented Language which is very robust. The same would be useful in building the new classes. Following are the features of Java :

- **Automatic Memory Management.**
- **Security**
- **Compiler / Interpreter Combo.**
- **Platform Independence.**
- **Threading.**

- Built-in Networking.

1.4.3 JSP

Java Server Pages (JSP) for short is Sun's solution for developing dynamic web sites. JSP provides excellent server side scripting support for creating database driven web applications. JSP enable the developers to directly insert java code into jsp file, this makes the development process very simple and its maintenance also becomes very easy. JSP pages are efficient, it loads into the web servers memory on receiving the request very first time and the subsequent calls are served within a very short period of time.

In today's environment most web sites servers dynamic pages based on user request. Database is very convenient way to store the data of users and other things. JDBC provide excellent database connectivity in heterogeneous database environment. Using JSP and JDBC its very easy to develop database driven web application.

Java is known for its characteristic of "write once, run anywhere".JSP pages are platform independent. One can port .jsp pages to any platform.

1.4.4 Tomcat

Apache Tomcat (or Jakarta Tomcat or simply Tomcat) is an open source servlet container developed by the Apache Software Foundation (ASF). Tomcat implements the Java Servlet and the JavaServer Pages (JSP) specifications from Sun Microsystems, and provides a "pure Java" HTTP web server environment for Java code to run. Features of Tomcat 6:

- memory leak prevention and detection
- protection against session fixation attacks
- a simple filter to add cross-site request forgery protection to an application
- simplified embedding
- alias support
- better security for the Manager and Host Manager applications
- and lots of internal code clean-up

1.4.5 Eucalyptus

Eucalyptus Tool Kit version 1.6.2 incorporates several new features and improvements as specified below:

- Deployment on multiple clusters
- Deployment of components (Cloud Controller,Walrus,Storage Controller,Cluster Controller)on different machines.

- **Enhanced maintenance support components are now "crash consistent", maintaining state across process restart or machine crash.**
- **Enhanced concurrency management: cloud requests are serviced asynchronously with minimal locking using eventual consistency for scale**
- **Support for open-source monitoring and health/status: Ganglia and Nagios interaction**
- **Re-engineered front-end Web-services stack to be faster and more robust**
- **Several fixes for S3 API compatibility**
- **Networking improvements, including multi-cluster support**
- **Improved hypervisor support**
- **Improved startup**
- **Web User Interface(UI) improvements including ability to select from themes (at compile time).**
- **Building and installation improvements**

1.4.6 KVM

KVM (for Kernel-based Virtual Machine) is a full virtualization solution for Linux on x86 hardware containing virtualization extensions (Intel VT or AMD-V). It consists of a loadable kernel module, kvm.ko, that provides the core virtualization infrastructure and a processor specific module, kvm-intel.ko or kvm-amd.ko. KVM also requires a modified QEMU although work is underway to get the required changes upstream.

Using KVM, one can run multiple virtual machines running unmodified Linux or Windows images. Each virtual machine has private virtualized hardware: a network card, disk, graphics adapter, etc.The kernel component of KVM is included in mainline Linux, as of 2.6.20. KVM is open source software. Eucalyptus uses KVM for virtualization.

Chapter 2

Literature Survey

This Chapter surveys on different types of Clouds and the work related done in Clouds. A good number of papers are referred and the Important points,Lacunae(if any) are highlighted from each paper.

2.1 Related Work in Clouds

Quite a few Research Clusters have worked in the related areas like Cluster, Grid and Cloud, addressing varied topics like Job Scheduling on virtual clusters [3, 4, 5, 6], but all of these focus on meeting the requirements of a single provisioning situation (either best effort or immediate but no ARs) except for Walters et al [6] and Borja Sotomayor et al [2, 8].

Resource management in Clouds is at a finer granularity and has many related levels. Managing resources dynamically and agilely in terms of the varied requirements of consumers is a challenge in Cloud Computing environments, and a Resource Allocation Strategy based on Market Mechanism (RAS-M) is proposed to settle this problem. RAS-M tries its best to achieve the equilibrium state through employing the present GA-based price adjusted algorithm [10].

OpenNebula is an open source virtual infrastructure manager that can be used to deploy virtualized services on both, a local pool of resources and external IaaS (Infrastructure as a Service) clouds. Haizea, a resource lease manager, can act as a Scheduling backend for OpenNebula providing features not found in other cloud software or virtualization-based datacenter management software, such as advance reservations and resource preemption [8].

2.2 Resource Leasing

Sr.No	1
Title	Resource Leasing and the Art of Suspending Virtual Machines
Author	Borja Sotomayor University of Chicago borja@cs.uchicago.edu Ruben Santiago Montero Ignacio Mart?n Llorente Universidad Complutense de Madrid rubensm,llorente @dacya.ucm.es Ian Foster University of Chicago Argonne National Laboratory foster@mcs.anl.gov
Source	2009 11th IEEE International Conference on High Performance Computing and Communications
Strength	Virtual Machines (VM) are the key technology to realize an Infrastructure as a Service (IaaS). It offers several benefits such as : Ability to securely partition physical servers To provide users with customized software environments.
Lacunae	Virtual Machines pose the problem of efficiently scheduling virtual machines on multi-host environments
Points	Model contributions are the following: • A model for predicting the runtime overhead of suspending and resuming VM-based leases (With potentially multiple VMs that must be co -scheduled within the lease) under a variety of conditions, removing many of the assumptions made in previous work. • Experimental results showing the degree of accuracy of the model in predicting the runtime overhead of suspending and resuming leases on a physical test bed. To run these estimations to schedule VM suspensions and resumptions. OpenNebula2 virtual infrastructure manager is used to run experiments on a physical test bed. • Simulation results showing the long-term effects of modifying parameters in the model, such as the amount of memory requested by VMs or the amount of network bandwidth available.

2.3 Resource Management in Heterogeneous Parallel Systems

Sr.No	2
Title	Stochastically Robust Resource Management in Heterogeneous Parallel Computing Systems
Author	BHoward Jay Siegel Colorado State University Department of Electrical and Computer Engineering Department of Computer Science Fort Collins, CO 80523-1373 email: HJ@ColoState.edu
Source	2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks
Strength	The stochastic robustness analysis approach can be applied to a variety of computing and communication system environments, including parallel, distributed, cluster, grid, Internet, cloud, embedded, multicore, content distribution networks, wireless networks, and sensor networks.
Lacunae	• Critical research problem is how to allocate resources to tasks to optimize some performance objective. • Systems frequently have degraded performance due to uncertainties, such as unexpected machine failures, changes in system workload, or inaccurate estimates of system parameters.

2.4 Desktop Cloud

Sr.No	3
Title	Volunteer Computing and Desktop Cloud: the Cloud@Home Paradigm
Author	Vincenzo D. Cunsolo, Salvatore Distefano, Antonio Puliafito and Marco Scarpa University of Messina, Contrada di Dio, S. Agata, 98166 Messina, Italy. Email: vdcunsolo,sdistefano,apuliafito,mscarpa@unime.it
Source	2009 Eighth IEEE International Symposium on Network Computing and Applications
Strength	• Compatibility limitations of Volunteer computing can be solved in Cloud Computing environments, allowing to share resources and services. • The Cloud@Home paradigm could be also applied to commercial Clouds, establishing an open computing-utility market where users can both buy and sell their services. • Cloud@Home both the commercial and the volunteer viewpoints coexist • Cloud@Home can be considered as the enhancement of the Grid-Utility vision of Cloud computing.
Lacunae	• The virtualization in Clouds implements the isolation of the services, but does not provide any protection from local access. • Interoperability is one of the most important goals of Cloud@Home and is an open problem in Grid, Volunteer and Cloud computing.

2.5 Enterprise Cloud

Sr.No	4
Title	Resource Management of Enterprise Cloud Systems Using Layered Queuing and Historical Performance Models
Author	David A. Bacigalupo, Jano van Hemert, Asif Usmani, Donna N. Dillenberger, Gary B. Wills and Stephen A. Jarvis School of Electronics and Computer Science, University of Southampton, SO17 1BJ, UK Data-Intensive Research Group, School of Informatics, University of Edinburgh, EH8 9AB, UK BRE Centre for Fire Safety Engineering, School of Engineering, University of Edinburgh, EH9 3JL, UK IBM T.J. Watson Research Centre, Yorktown Heights, New York, 10598, USA High Performance Systems Group, Department of Computer Science, University of Warwick, CV4 7AL, UK
Source	2010 IEEE
Points	Two common approaches used in the literature for making the response time predictions are extrapolating from historical performance data and solving queuing network models. Examples of the first approach include the use of both coarse and fine grained historical performance data. The former involves recording workload information and operating system/database load metrics, and the latter involves recording the historical usage of each machine's CPU, memory and IO resources by different classes of workload.

2.6 Cost-Effective Resource Leases

Sr.No	5
Title	Enabling Cost-Effective Resource Leases with Virtual Machines Computing
Author	Borja Sotomayor, Kate Keahey, Ian Foster, Tim Freeman
Source	Borja Sotomayor site
Lacunae	Achieving stricter lease semantics (Advance reservation) is difficult because it often lead to utilization problems in the scheduler caused by the need to "drain" jobs off of a group of resources before the job reservation starts.
Points	• Applications consisting of workflows of small tasks (such as Montage, GADU can be more efficiently scheduled by a workflow engine (e.g Pegasus or Swift) when using leased resources than when each request must pass via a traditional scheduler . • making leasing more cost-effective, one can support a multi-level scheduling model, by decoupling resource provisioning from execution management. • VMs are used to make on-demand short-term leasing of resources cost effective. This architecture allows resource providers to satisfy short-term leasing requests while continuing to support existing workloads (i.e., batch processing). • Virtualization can achieve improved performance, both from the provider's perspective (throughput) and the user's perspective (running time), even in the presence of overhead incurred by using VMs. • By using the suspend/resume capability of virtual machines, batch computations and timer-driven leasing requests (such as advance reservations) can be interleaved in such a way that resources do not have to be backfilled and "drained" before the start of a lease resulting in improved resource utilization.

2.7 Market Mechanism

Sr.No	6
Title	RAS-M:Resource Allocation Strategy based on Market Mechanism in Cloud Computing
Author	Xindong YOU, Xianghua XU, Jian Wan, Dongjin YU School of Computer Science and Technology Hangzhou Dianzi University Hangzhou, China
Source	2009 Fourth China Grid Annual Conference
Strength	Demand and Supply is balanced nearly, which validates RAS-M is effective and practicable, and is capable of achieving its goal.
Points	• Cloud is a type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service level agreements established through negotiation between the service provider and consumers. • Employing the virtualized technologies, such as VMWare PC, VMWare ESX, Xen, and KVM etc, one or more VMs can run on a physical machine simultaneously • RAS-M tries it best to achieve the equilibrium state through employing the present GA(Genetic Algorithm)-based price adjusted algorithm.

2.8 Market-Oriented Cloud Computing

Sr.No	7
Title	Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities.
Author	Rajkumar Buyya1,2, Chee Shin Yeo1, and Srikumar Venugopal1 1 Grid Computing and Distributed Systems (GRIDS) Laboratory Department of Computer Science and Software Engineering The University of Melbourne, Australia Email: raj, csyeo, srikumar@csse.unimelb.edu.au Manjrasoft Pty Ltd, Melbourne, Australia
Source	The 10th IEEE International Conference on High Performance Computing and Communications
Lacunae	Issues are • How the participants in a market can obtain restitution in case an SLA is violated.
Points	• Cloud computing which promises reliable services delivered through next-generation data centers that are built on compute and storage virtualization technologies. • The Cloud appears to be a single point of access for all the computing needs of consumers. • A Cloud is a type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers.

2.9 Cloud Comparison

Sr.No	8
Title	Comparison of Several Cloud Computing Platforms
Author	Junjie Peng School of computer science and High performance computing center Shanghai University Shanghai, 200072 P.R. China Xuejun Zhang Qinghua machinery factory at Changzhi Changzhi, 046012 P.R China Zhou Lei, Bofeng Zhang, Wu Zhang, Qing Li School of computer science and High performance computing center Shanghai University Shanghai, 200072 P.R. China
Source	Second International Symposium on Information Science and Engineering

2.10 Various Cloud Platforms

Characteristic	Abicloud	Eucalyptus	Nimbus	OpenNebula
Cloud Character	Public / Private	Public	Public	Private
Scalability	Scalable	Scalable	Scalable	Dynamically Scalable
Cloud Form	Iaas	Iaas	Iaas	Iaas
Compatibility	Does not support EC2	Does support EC2, S3	Does support EC2	Open,Multi-Platform
Deployment	Pack and redeploy	Dynamical Deployment	Dynamical Deployment	Dynamical Deployment
Deployment manner	Web Interface Drag	Commandline	Commandline	Commandline
Transplant ability	Easy	Common	Common	Common
VM Support	VirutalBox, Xen, VMware, VM	Xen, VMware, KVM	Xen	Xen, VMware
Web Interface	Libvirt	Web Service	EC2, WSDL, WSRF	Libvirt,EC2,OCCI, API
Structure	Open Platform encapsulate core	Module	Lightweight Components	Module
Reliability	-	-	-	Rollback host and VM
OS Support	Linux	Linux	Linux	Linux
Development Language	Ruby, C++ ,Python	Java	Java,Python	Java

2.11 Summary

This chapter describes some recent works related to advance reservation in Networks Grids and Clouds. This chapter also focuses on the significant points discussed in several standard papers specified in the above section. A main focus on the different cloud platforms with various characteristics is also present at the end of this chapter.

Chapter 3

Reservation States

Advance reservation (AR) is the process of requesting resources for use at specific times in the future . Common resources that can be reserved or requested are compute nodes (CNs), storage elements (SEs), network bandwidth or a combination of any of those, as mentioned earlier.

3.1 States of a Reservation

An Advance Reservation request can be in one of several states during its lifetime as shown in Figure 3.1 Transitions between the states are defined by the operations that a consumer performs on the reservation. These states are defined as follows:

3.1.1 Request

When a request for a reservation of Resources is first made, it is in the Initial state.

3.1.2 Reject

The reservation is not fruitfully allocated; cause may be the existing reservation has expired or the slots may be full.

3.1.3 Accept

A request for a new reservation has been agreed upon.

3.1.4 Commit

A reservation has been confirmed by a consumer before the expiry time and will be privileged by the requested Resource.

3.1.5 Change Request

A consumer is trying to alter the requirements for the reservation prior to its starting. If it is successful, the reservation is committed with the new requirements; otherwise, the parameters remain the same. For instance, one can increase the bandwidth that has already been requested. A modification that reduces its requirements normally succeeds,

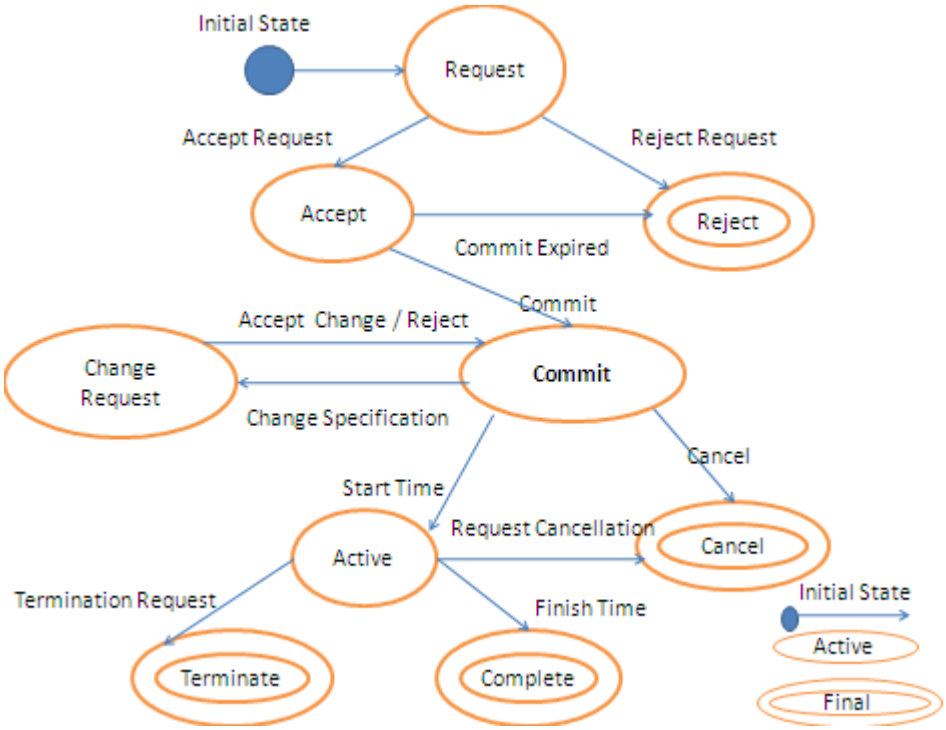

Figure 3.1: A State Transition diagram depicting the states for Advance Reservation

although certain factors may cause reduction modifications to fail, such as local policy that does not allow small reservations on some resources. In no case should the underlying system implement a modification such that if the modification fails, the original reservation is lost. For example, a simple implementation may implement modification by canceling a reservation and making a new reservation, but if the new reservation fails and it cannot be rolled back to the original reservation, this would be undesirable.

3.1.6 Active

The system executes the reservation as the start time has been reached.

3.1.7 Cancel

A consumer may cancel the reservation as it may no longer be required.

3.1.8 Complete

The reservations end time has been reached.

3.1.9 Terminate

A consumer terminates an active reservation before the end time is reached.

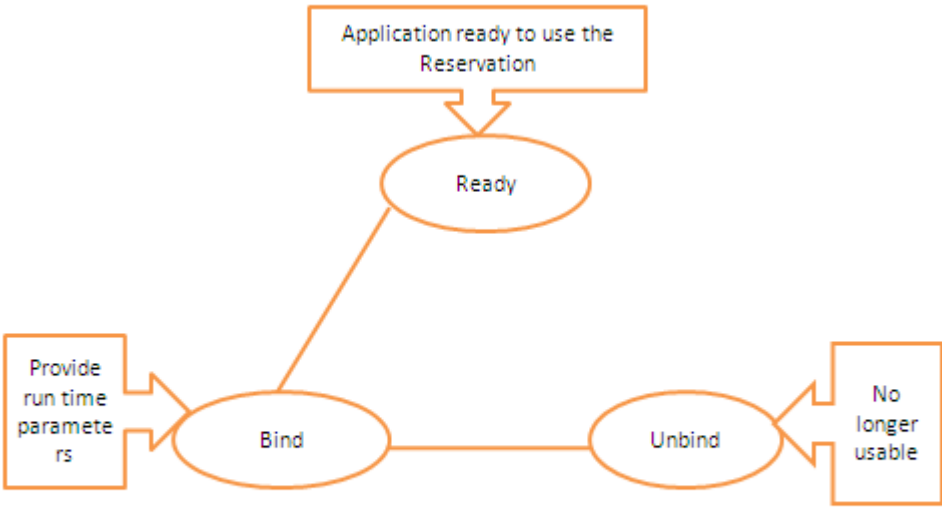

Figure 3.2: A State Transition diagram depicting Bind and Unbind

3.1.10 Bind Reservation

When the application is ready to use a reservation, it may need to provide run-time information that was not available at the time the reservation was made as shown in Figure 3.2. This is known as binding a reservation [9]. For example, network reservations require port numbers to be specified, but they are not usually known at reservation time. Not all reservations require such run-time parameters.

3.1.11 Unbind Reservation

A reservation can be unbound. It then will no longer be usable by the person using the reservation as shown in Figure 3.2. It can be rebound, however with new run-time parameters as shown in Figure 3.2

3.2 Query Reservation Status

Consumer can query for the status of a reservation by polling it as shown in Figure 3.4. The status includes whether the start of the reservation has begun and whether the reservation has been committed .

3.2.1 Query Reservation Attributes

Consumer can query for attributes associated with an existing reservation. These include begin and end time of the given reservation and whether it is a two-phase commit

Figure 3.3: A State Transition diagram depicting Bound and Rebound

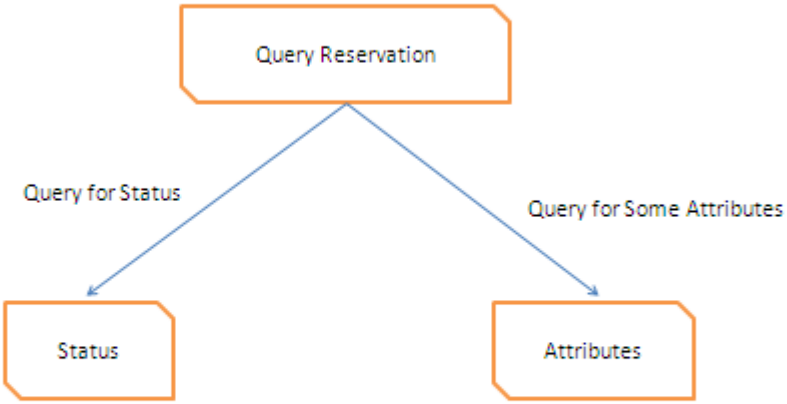

Figure 3.4: Query a Reservation

reservation. The attributes also include specific information required to actually use a reservation. For example, attributes are a Folder name where data was staged on or a Queue name which has to be used for submitting a job as shown in Figure 3.1

3.3 Register Callback

One can provide a function that will be called when the status of a reservation changes or when the reservation manager wishes to provide extra information to the application. This information may include notification that the related reservation appears to be too small.

3.4 Summary

This chapter discusses the various states a parallel job submitted by the Consumer can be in. The parallel job is submitted by the Consumer in Advance. Finally, after the reservation has been made and the Consumer can submit some parameters at run time called as Bind a Reservation. The Consumer can query for the details later when required. The details are stored in his/her directory.

Chapter 4

Advance Reservation Data Structures and Operations

4.1 Introduction

In order to reserve available resources in a Cloud, a Consumer must first submit a request by specifying a series of parameters such as number of compute nodes (CNs) needed,start time and duration of his/her jobs. Then, the Advance Reservation system checks for the feasibility of this request. If there are no available nodes for the requested time period, the request is rejected. Consequently, the Consumer may resubmit a new request with a different start time and/or duration until available nodes can be found. Given this scenario, the choice of an efficient data structure can significantly minimize the time complexity needed to search for available compute nodes, add new requests, and delete existing reservations.

4.2 Basic Operations

An efficient data structure for managing reservations plays an important role in order to minimize the time required for searching available resources, adding, and deleting reservations.An efficient data structures for managing the Advance Reservation is proposed in this section called Eucalyptus Cloud Advance Reservation Queue(ECARQ). The proposed data structure has buckets with a fixed delta, which represents the smallest slot duration or slot interval as shown in Figure 4.1. Each bucket contains mrv (the number of already reserved CNs in this bucket) and a linked list (sorted or unsorted), containing the reservations which start in this time bucket. For enabling a fast O(1) access to a particular bucket, the following is used formula:

$$i = [t/delta] mod M \tag{4.1}$$

where i is the bucket index,delta the smallest slot interval or slot duration, t is the request time, and M is the number of buckets in the data structure.

A common input for the operations is the tuple (t_{start}, t_{end}, numCN)here :

- t_{start} is the Reservation start time
- t_{end} the Reservation end time and
- numCN is the number of Computer Nodes required for a Consumer at the specified interval.

In order to support Advance Reservation in Clouds, ECARQ data structure needs to perform the following basic operations:

- Search : checking for availability of CNs in a given time interval. This operation is defined as searchReserv(t_{start}, t_{end}, numCN)
- Insert: inserting a new reservation request into the ECARQ data structure. This operation is performed only when the previous search phase succeeded. For addition, the new reservation is represented as addReserv((t_{start}, t_{end}, numCN; CNM) where CNM is an Consumer object storing the Consumer's jobs and other relevant information.
- Delete: removing the existing reservation from the ECARQ data structure. This operation is conducted only when the add phase succeeds and the reservation's end time passed. It is described as deleteReserv((t_{start},t_{end},numCN)s shown in 4.1.
- Interval Search: the ECARQ data structure finds an alternative time for a rejected request. This operation helps Consumer whose requests got rejected in negotiating a suitable reservation time. Therefore, the performance of this operation also needs to be considered when choosing the appropriate data structure. It is defined suggestInterval((t_{start},t_{end},numCN)

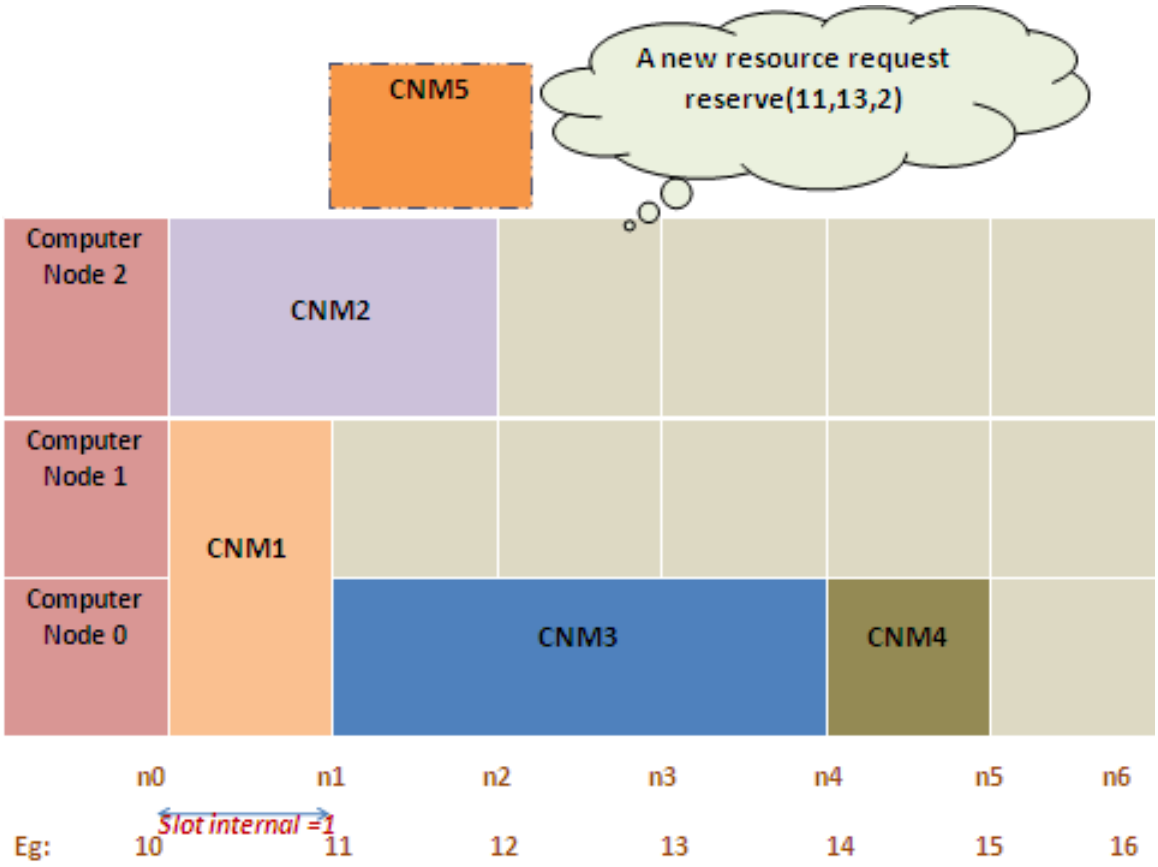

Figure 4.1: An example of Advance Reservations for reserving Computer Nodes. The maximum available computer nodes is 3. A dotted box denotes a new request.

Figure 4.1 shows an example of existing reservations represented in a time-space diagram. When a new request from Consumer 5(CNM5) arrives, the resource scheduler checks for any available Computer Nodes. In the above example, the request is defined as reserv((t_{start},t_{end},numCN) with numCN = 2. However, only one node is available, hence, this request will be rejected. By performing suggestInterval(11; 16; 2) on this request, the system manages to find the next available time, which is from time 13 to time 15. Note that in this example, the ending time has been increased for a bigger search time range.

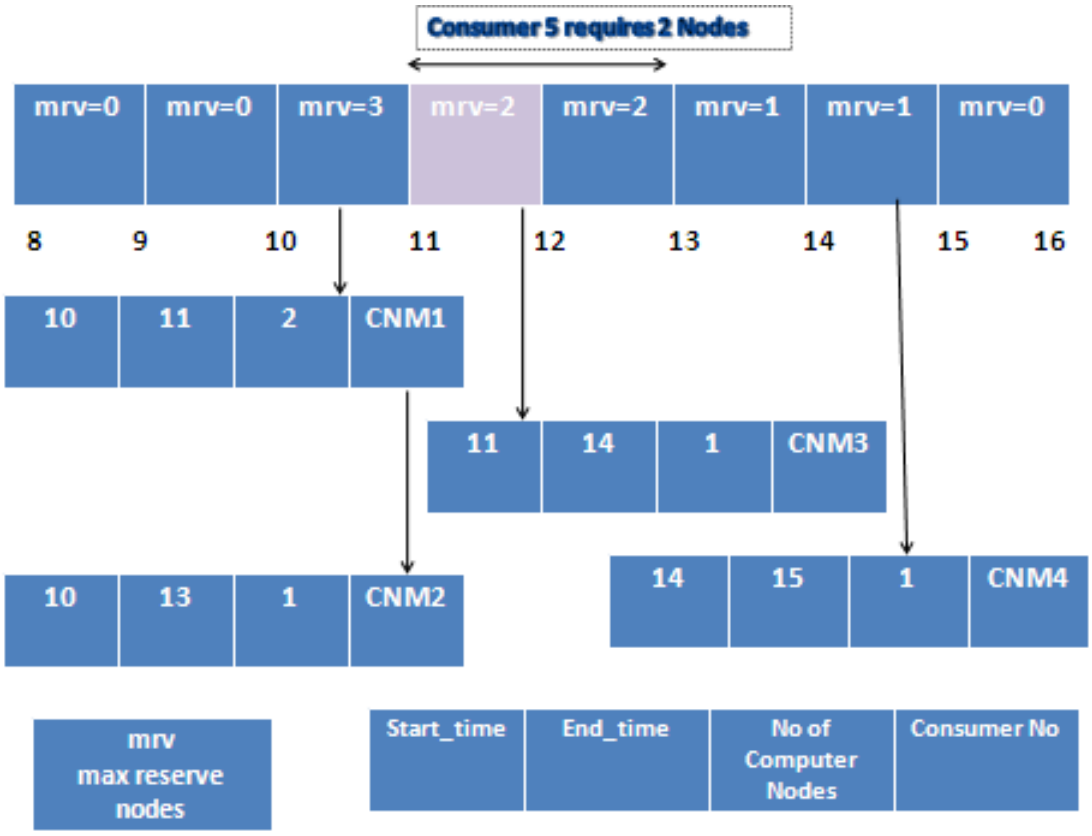

Figure 4.2: A representation of storing reservations in ECARQ with Sorted Queue and N = 1. A request from Consumer 5 is rejected because not enough Computer Nodes for slot [11, 12] as shown by the shaded box.

4.3 Proposed Advance Reservation Algorithms

This section discusses the proposed algorithms for performing the different operations like Insert, Delete, SearchInterval and SuggestInterval ,on the proposed Data Structures ECARQ.

Algorithm 1 searchReservation((t_{start},t_{end},numCN) in ECARQ

1: $start\_bucket \leftarrow get\_bucket\_index(t_{start})$ {get the starting index.}
2: $end\_bucket \leftarrow get\_bucket\_index(t_{end})$ {get the ending index.}
3: $finish \leftarrow 0$
{a case where it needs to wrap around the array.}
4: **if** *end_bucket* <start_bucket **then**
5: $finish \leftarrow M$
6: **else**
7: $finish \leftarrow end\_bucket$
8: **end if**
9: **for** i =start_bucket **to** finish **do**
10: **if** i ==M **then**
11: $i \leftarrow$0 {set to first index.}
12: $finish \leftarrow end\_bucket$
13: **end if**
14: **if** (*bucket*[i].rv + numCN) ¿ maxCN **then**
15: **return false**
16: **end if**
17: **end for**
18: **return true**{slot is full.}

Algorithm 2 addReservation((t_{start},t_{end},numCN) in ECARQ

1: $start\_bucket \leftarrow get\_bucket\_index(t_{start})$ {get the starting index.}
2: $end\_bucket \leftarrow get\_bucket\_index(t_{end})$ {get the ending index.}
3: $bucket$[start_bucket].addInfo(Client) {stores users job and other details.}
4: $finish \leftarrow 0$ {a case where it needs to wrap around the array.}
5: **if** end_bucket <start_bucket **then**
6: $finish \leftarrow M$
7: **else**
8: $finish \leftarrow end\_bucket$
9: **end if**
10: **for** i =start_bucket **to** finish **do**
11: **if** i ==M **then**
12: $i \leftarrow 0$ {set to first index.}
13: $finish \leftarrow end\_bucket$
14: **end if**
15: bucket[i].$rv \leftarrow$ $bucket$[i].rv + numCM
16: **end for**

Algorithm 3 suggestInterval((t_{start},t_{end},numCN) in ECARQ

1: $start\_bucket \leftarrow get\_bucket\_index(t_{start})$ {get the starting index.}
2: $end\_bucket \leftarrow get\_bucket\_index(t_{end})$ {get the ending index.}
3: $tot\_req \leftarrow 1 + end\_bucket - start\_bucket$ {total slots required.}
4: $new\_start \leftarrow start\_bucket$ {the new starting index}
5: $count \leftarrow 0$
6: $last\_bucket \leftarrow get\_bucket_indx$(t_start + MAX_LIMIT) {the last bucket to Search}
7: $finish \leftarrow M$ { a case where it needs to wrap around the array.}
8: **if** $last\_bucket < start\_bucket$ **then**
9: $finish \leftarrow$M
10: **else**
11: $finish \leftarrow last\_bucket$
12: **end if**
13: **for** i =start_bucket **to** finish **do**
14: **if** i ==M **then**
15: $i \leftarrow 0$ {set to first index.}
16: $finish \leftarrow last\_bucket$
17: **end if**
18: **if** (bucket[i].$rv + numCN$) >maxCN **then**
19: $new\_start \leftarrow i + 1$
20: $count \leftarrow 0$
21: **else**
22: $count \leftarrow count + 1$
23: **end if**
24: **end for**

Deleting an existing reservation applies to the same principle as adding a new one. It can be done by removing the reservation from the starting bucket and decrementing mrv through out the relevant bucket interval.

4.4 Advantages of ECARQ

A lot many advantages exists in ECARQ which is like an Array based Data Structure. The advantages are explained as follows:

- potential to add new reservations directly into a particular bucket. Thus it has a quick O(1) access to the bucket.
- potential to reuse these buckets for the next time period.
- built only once in the beginning.
- easy to search and compare by using iteration
- flexibility in handling resource availability.
- In Clouds, Computer Nodes(CN) can be added or removed periodically and dynamically. Such issues can be levered by a reservation system or a resource scheduler by setting the amount of available CNs on that resource appropriately. Furthermore, existing reservations can be relocated to other CNs through the add and delete operations.

4.5 Summary

This chapter discusses a proposed data structure called ECARQ for managing reservations. It is very efficient and plays an important role in order to minimize the time required for searching available resources, adding, and deleting reservations. The Algorithms which operate on this Data Structures are also discussed in this chapter.Also there are highlights on Advantages of the proposed Data Structures.

Chapter 5

Eucalyptus

5.1 Introduction

Eucalyptus enables the creation of on-premise private clouds, with no requirements for retooling the organization's existing IT infrastructure or need to introduce specialized hardware. Eucalyptus implements an IaaS (Infrastructure as a Service).Eucalyptus cloud to be turned into a hybrid cloud, capable of drawing compute resources from public cloud. And Eucalyptus is compatible with a wealth of tools and applications that also adhere to the de facto EC2 and S3 standards.

5.2 Eucalyptus Components

Each Eucalyptus service component exposes a well-defined languageagnostic API in the form of a WSDL document containing both the operations that the service can perform and the input/output data structures. Inter-service authentication is handled via standard WS-Security mechanisms. There are five high-level components, each with its own Web-Service interface, that comprise a Eucalyptus installation Figure 6.1. A brief description of the components within the Eucalyptus system follows.

5.2.1 Cloud Controller

Cloud Controller (CLC) is the entry-point into the cloud for administrators, developers, project managers, and end users. The CLC is responsible for querying the node managers for information about resources, making high level scheduling decisions , and implementing them by making requests to cluster controllers. The CLC, as shown in Figure 6.1, is also the interface to the management platform. In essence, the CLC is responsible for exposing and managing the underlying virtualized resources (servers, network, and storage) via a well-defined industry standard API (Amazon EC2) and a Web-based user interface.

5.2.2 Cluster Controller

Cluster Controller (CC) generally executes on a cluster front-end machine, or any machine that has network connectivity to both the nodes running NCs and to the machine running the CLC. CCs gather information about a set of VMs and schedules VM execution on specific NCs. The CC also manages the virtual instance network and participates in the

enforcement of SLAs as directed by the CLC. All nodes served by a single CC must be in the same broadcast domain (Ethernet).

5.2.3 Node Controller

Node Controller (NC) is executed on every node that is designated for hosting VM instances. NCs control the execution, inspection, and termination of VM instances on the host where it runs, fetches and cleans up local copies of instance images (the kernel, the root file system, and the ramdisk image), and queries and controls the system software on its node (host OS and the hypervisor) in response to queries and control requests from the cluster controller. The Node controller is also responsible for the management of the virtual network endpoint.

5.2.4 Storage Controller

Storage Controller (SC) implements block-accessed network storage (e.g. Amazon Elastic Block Storage(EBS) and is capable of interfacing with various storage systems (NFS, iSCSI, etc.). An elastic block store is a Linux block device that can be attached to a virtual machine but sends disk traffic across the locally attached network to a remote storage location. An EBS volume cannot be shared across instances but does allow a snap-shot to be created and stored in a central storage system such as Walrus, the Eucalyptus storage service.

5.2.5 Walrus(put/get storage)

Walrus (put/get storage) allows users to store persistent data, organized as eventually-consistent buckets and objects. It allows users to create, delete, list buckets, put, get, delete objects, and set access control policies. Walrus is interface compatible with Amazons S3, and supports the Amazon Machine Image (AMI) image-management interface, thus providing a mechanism for storing and accessing both the virtual machine images and user data.

5.2.6 Management Platform

Management Platform provides an interface to various Eucalyptus services and modules. These features can include VM management, storage management,user/group management, accounting, monitoring , SLA definition and enforcement, cloud-bursting, provisioning, etc.

5.2.7 ARM

Advance Resource Management (ARM), the Resource provisioning model, is the model that is responsible for Advance Reservation. It interacts with CC,NC,SC and CLC before confirming the reservation to the consumer. It also provides support for immediate and best-effort provisioning.

5.3 Eucalyptus Configuration

With these components, Eucalyptus can be configured to support a wide variety of infrastructure features and topologies. For example, four different networking modes are supported , each corresponding to a different level of security and infrastructure intrusiveness allowing system administrators to tune each cloud configuration to meet local policy and management needs. It is also possible to deploy Eucalyptus to include different hypervisors and virtualization technologies within a unified cloud exporting a single API . Thus, a Eucalyptus cloud can act as a platform for unifying a variety of technologies (each at a potentially different point in its data center lifecycle) within a single cloud.

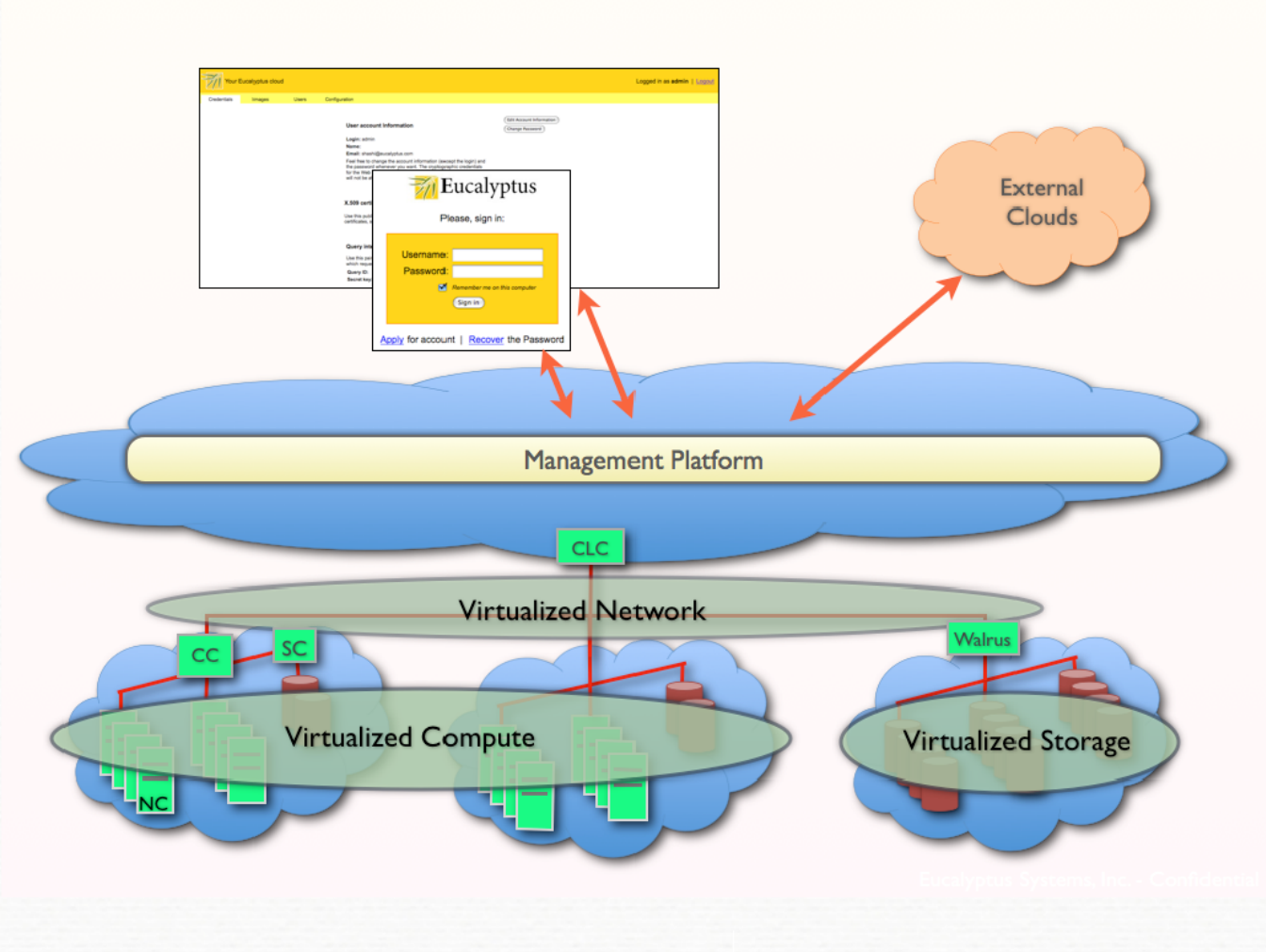

Figure 5.1: Conceptual Representation of Eucalyptus Cloud

5.4 Advance Reservation Data Structure

This section confers about the flow in which the AR Module would initiate and initialize the Data Structures. The Figure 6.2 depicts that first "Ubuntu Enterprise Cloud" (UEC) starts,then the Eucalyptus and then the AR Data Structures are initialized which gets its values from the configuration file eucalyptus.conf [22] which is used to configure and customize Eucalyptus. The following options are read from the above specified file:

- VNET-MODE="MANAGED-NOVLAN" There are four modes to choose from (MANAGED, MANAGED-NOVLAN, SYSTEM, or STATIC) and each has its own sub-options. The first modes (MANAGED, MANAGED-NOVLAN) configure eucalyptus to fully manage the VM networks, and enables the ability to use security groups and dynamic public IP assignment (with and without vlan tagging of security group networks, respectively).
- NODES="" The list of Node Controllers the Cluster Controller will communicate with.
- MAX-CORES="" The maximum number of CPU/cores Eucalyptus is allowed to use on the node (at the moment there is no difference between cores and CPU). If this is commented out, Eucalyptus will use all available CPU/cores it can find. Default value is 2.

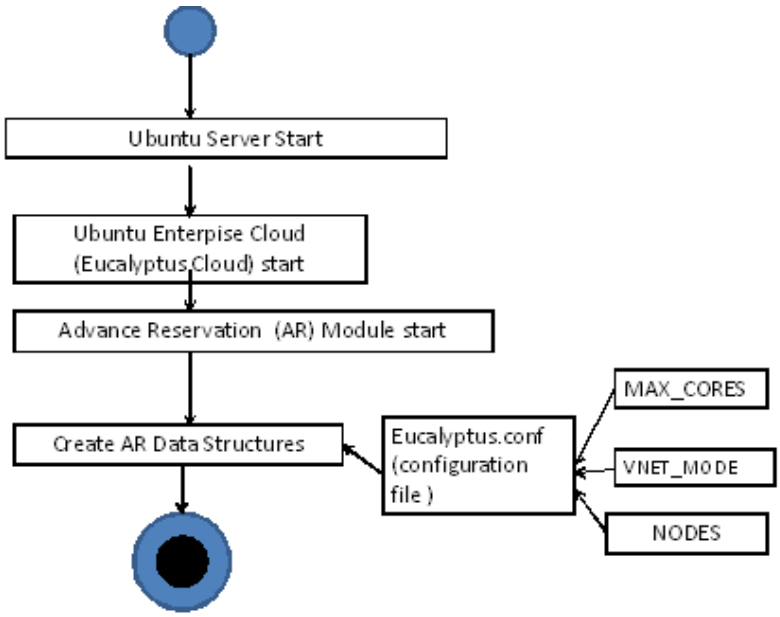

Figure 5.2: A State diagram to depict the initialization of Advance Reservation Data Structure.

5.5 Cloud Configuration

Steps to install Ubuntu Cloud:

- Prerequisites

  Following the system configuration required for a machine on which Eucalyptus cloud would be installed. The suggestions listed in the following table are as specified in the community URL of Ubuntu as shown in Figure 5.3[24].

- Install the Cloud/Cluster/Storage/Walrus Front End Server. When the CD is booted, select "Install Ubuntu Enterprise Cloud" (UEC) as shown below in Figure 5.4.

- The installer will detect if any other Eucalyptus components are present as shown below in Figure 5.6 . One can then choose the following components as shown in Figure5.7.

- Eucalyptus is made of 5 types of components that can live on the same or on separate systems . The CLC (Cloud controller) is the entry point to the cloud, which is made of one or more clusters as shown below in the Figure5.5.

- The Walrus is a unique component providing an S3-like service in the cloud.

- A CC (Cluster controller) controls a given cluster.

- A SC (Storage controller) handles storage in a given cluster.

- A NC (Node controller) handles VMs in a given cluster. There can (and should) be multiple NCs in each cluster.

Parent components need to register child components before they can use them as part of the cloud. CLC is the parent for CCs, SCs and Walrus child components, while the CC is the parent for NC child components.

Hardware	Minimum	Suggested	Notes
CPU	1GHz	2 x 2GHz	for an all-in-one front end, it helps to have at least a dual core processor
Memory	2GB	4GB	the Java web front end benefits from lots of available memory
Disk	5400rpm IDE	7200rpm SATA	slower disks will work, but will yield much longer instance startup times
Disk Space	40GB	200GB	40GB is only enough space for only a single image, cache, etc., Eucalyptus does *not* like to run out of disk space.
Networking	100Mbps	1000Mbps	machine images are hundreds of MB, and need to be copied over the network to nodes

Figure 5.3: System configuration required for a machine with Eucalyptus Cloud to be Configured.

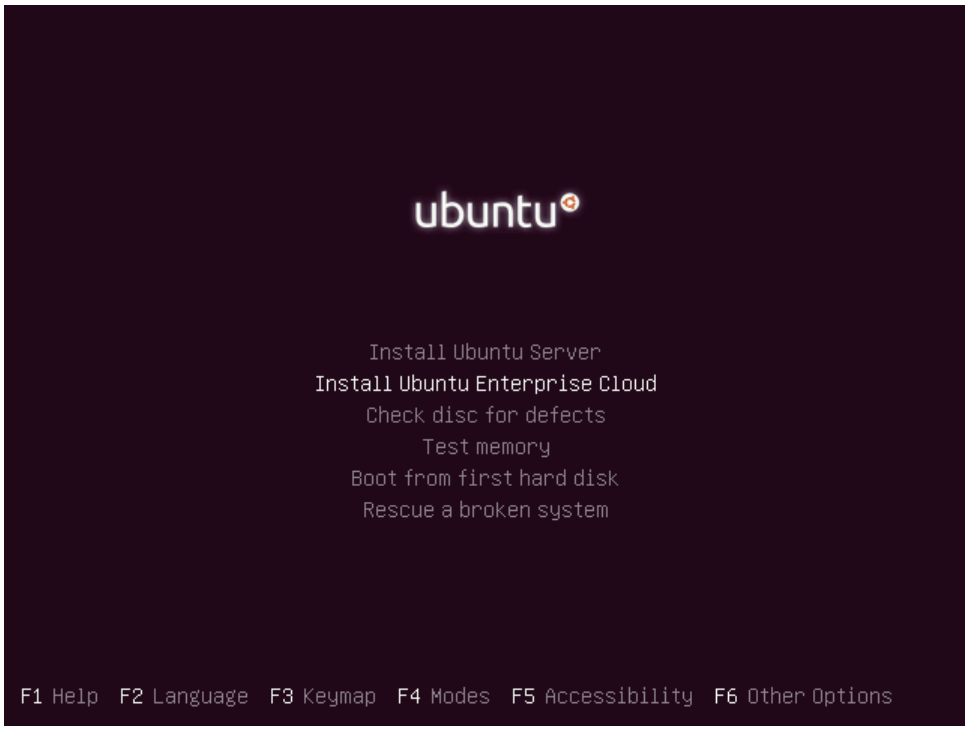

Figure 5.4: Boot screen of Ubuntu Enterprise Cloud (UEC).

Name the Cluster that is to be installed on the machine as shown in Figure 6.7

A range of IP(Internet Protocol) addresses on LAN(Local Area Network) that the cloud can be allocated to instances. e.g. 192.168.1.200 - 192.168.1.249. Once all the components are installed the services could be started using the following commands:

- sudo service eucalyptus-cloud start.This command is used to start the eucalyptus cloud controller.
- sudo service eucalyptus-cc start.This command is used to start the eucalyptus cluster controller.
- sudo service eucalyptus-nc start. This command is used to start the eucalyptus node controller.
- sudo service eucalyptus-walrus start. This command is used to start the eucalyptus walrus service.

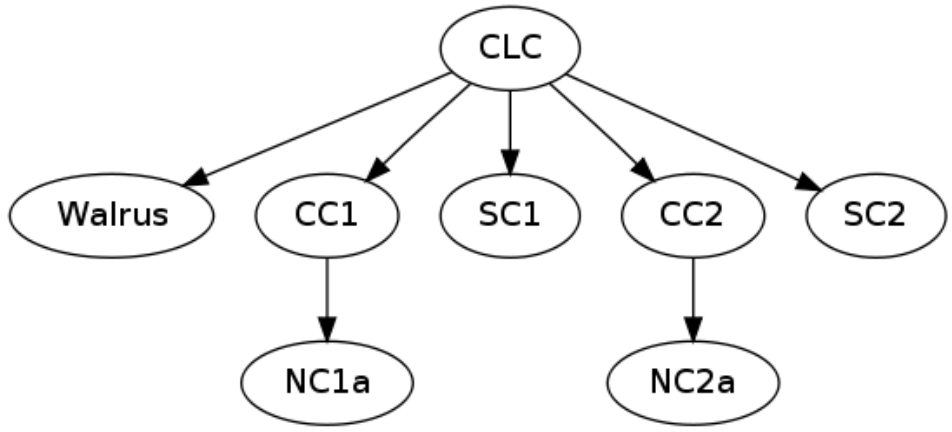

Figure 5.5: A Tree Structure showing the components of Cloud.

- sudo service eucalyptus-sc start. This command is used to start the eucalyptus storage controller service.

If all these services are running its an indication that the cloud has been installed successfully.

5.6 Summary

This chapter discuses about Eucalyptus cloud ,the Components and the Installations steps.All the screens which come during the installations are discussed in this chapter.Also discussions on commands to start the cloud Services on Ubuntu is done.

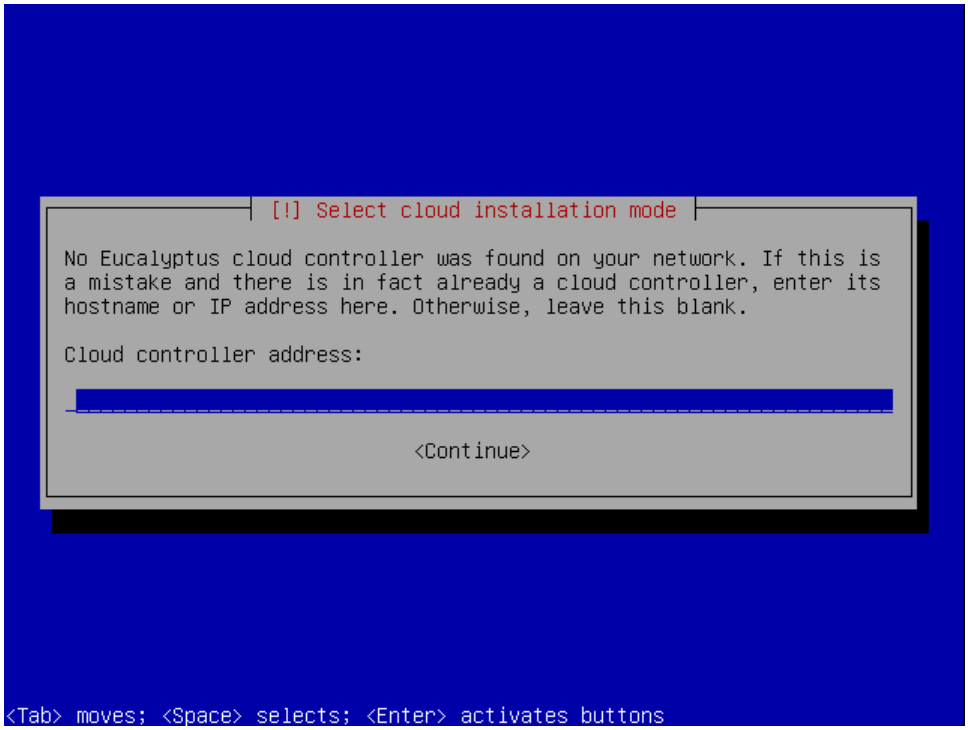

Figure 5.6: Screen to Input the IP Address of Cloud Controller.

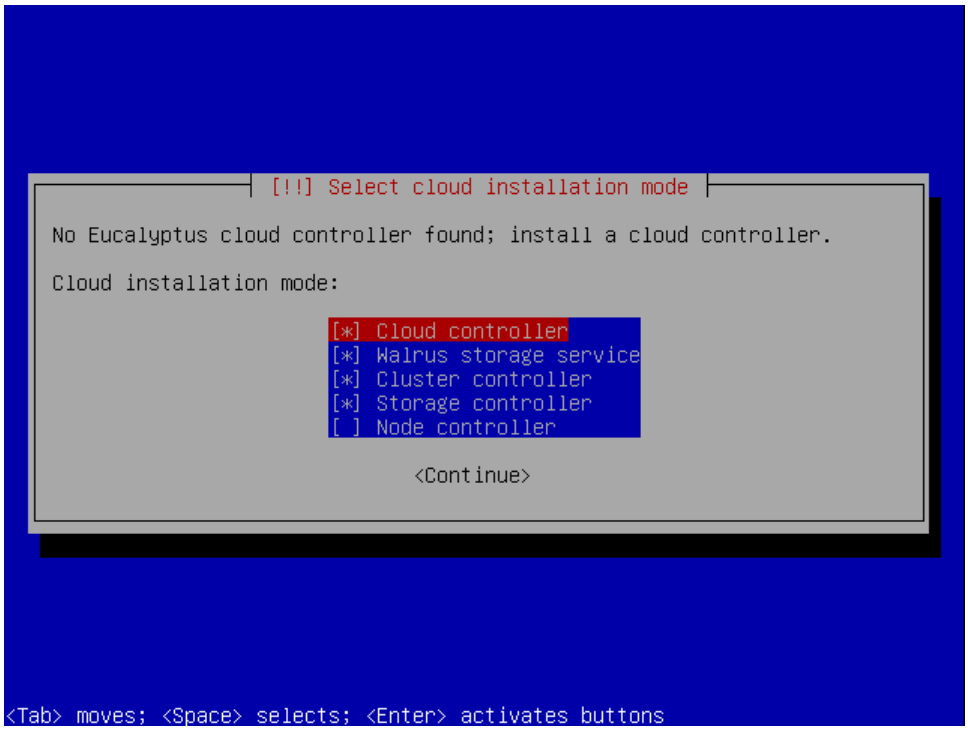

Figure 5.7: Screen depicting the selection of Cloud Controller Component of Eucalyptus.

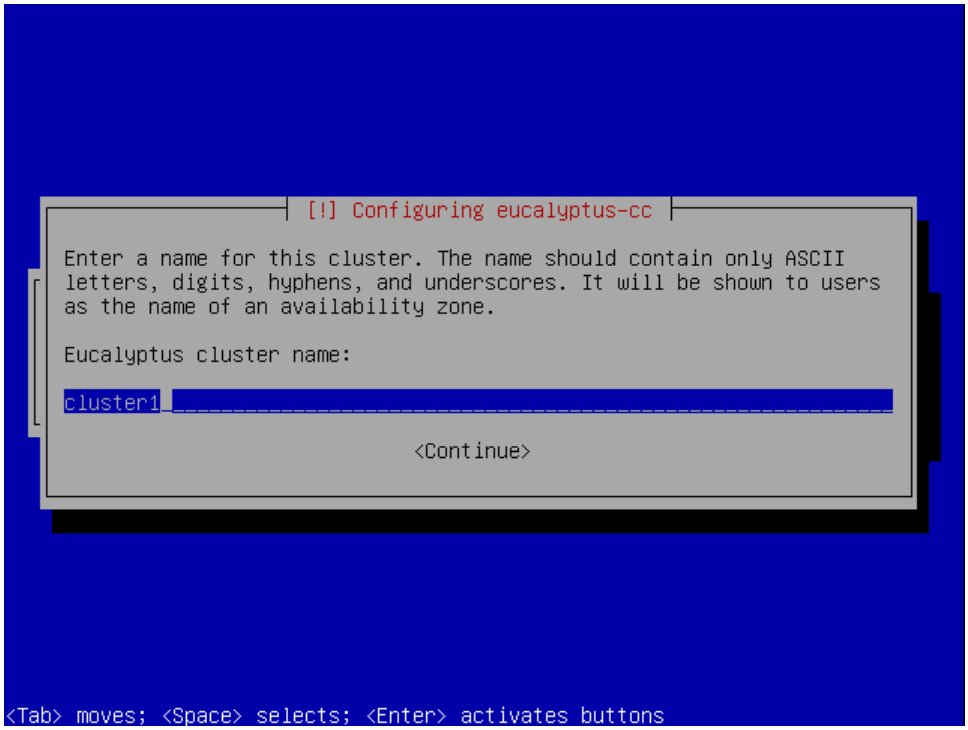

Figure 5.8: Screen to Input the Eucalyptus Cluster-name.

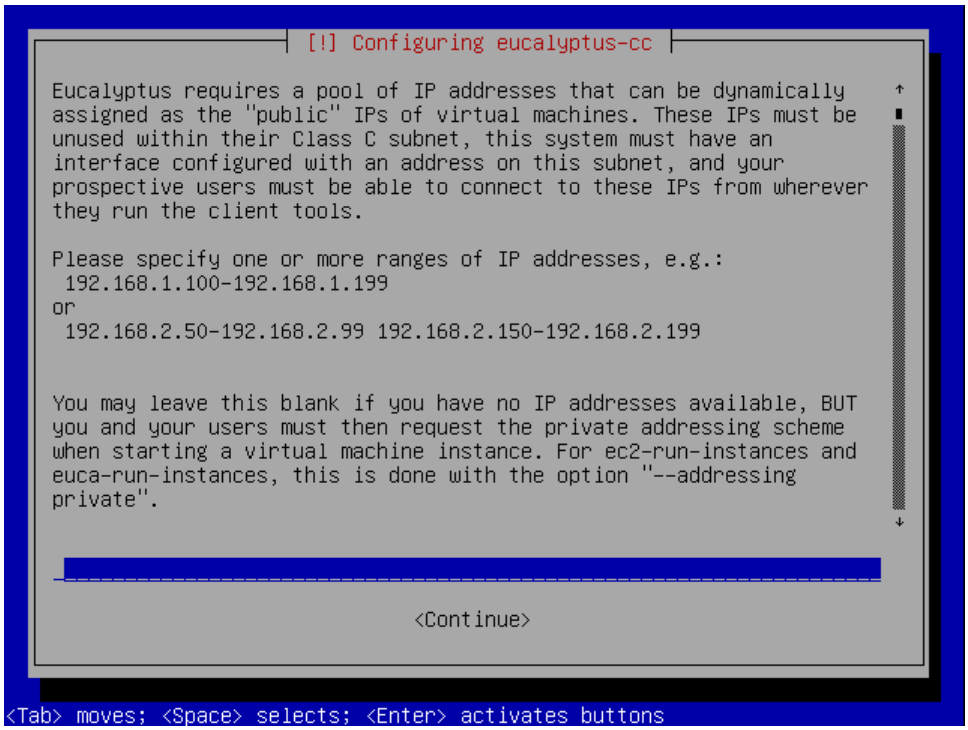

Figure 5.9: Screen to Input the IP Adress 192.168.1.1.

Chapter 6

Service Offered and InterFaces

6.1 Introduction

Cloud computing is complete new technique put forward from industry circle, it is the development of parallel computing, distributed computing and grid computing, and is the combination and evolution of virtualization, utility computing, Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) Software-as-a-Service (SaaS) and a term coined by us Compilers as a Service (CaaS).

Cloud computing can provide many kinds of service modes, including IaaS, PaaS and SaaS. Where SaaS means the service provided to client is the applications running on the cloud computing infrastructure provided by the service providers. It can access by thin client interfaces such as bowser etc. PaaS refers to deploy the Applications created by the development language and tool say Java, python, .Net etc. provided by the service providers to the cloud infrastructure. IaaS refers to the services provided to the users is to lease the processing power, storage, network and other basic computing resources, with which users can deploy and run any software including operating systems and applications. To all these services, there is no need for users to manage or control the cloud infrastructure, including network, server, operating system, storage and even the functions of Applications.CaaS means the service providers accept the parallel jobs from the Consumers who send these parallel jobs at a pre-selected time by using the Web based Interface.

6.2 Services Offered

The Services offered by the Cloud are CaaS on top of Infrastructure as a Service (IaaS) as depicted in Figure 6.1.

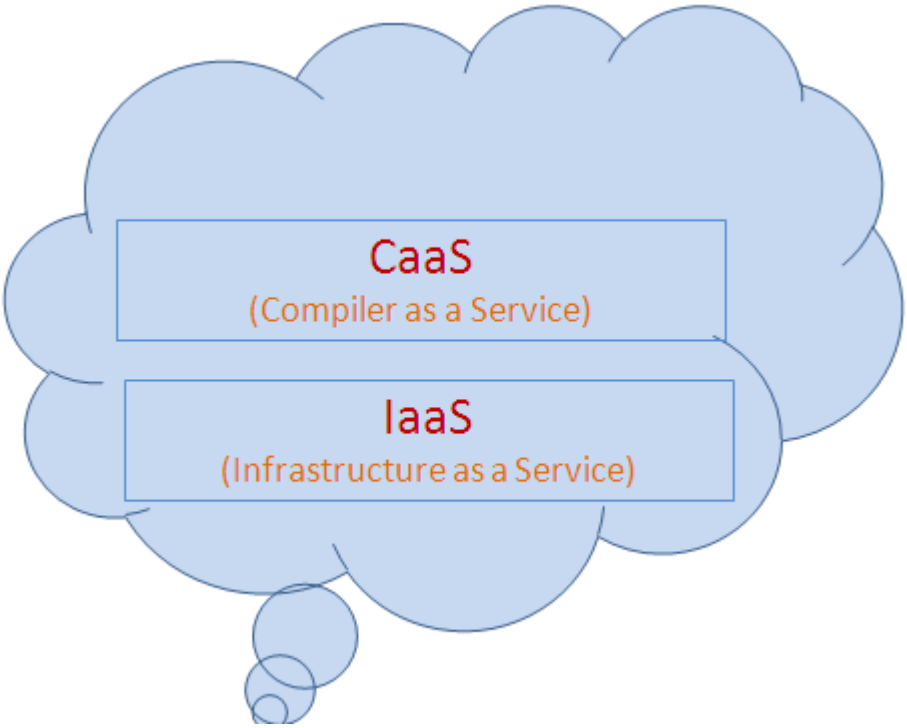

Figure 6.1: Services offered by the Cloud.

A Login screen for different types of users is depicted in Figure 6.2.User Name and password needs to be entered for authentication.Two specific types of users an Administrator and Consumers (Service Utilizers)are using the same screen. An Interface for new Consumer is also provided where they can register themselves for utilizing the services.

A Change Password Screen as depicted above in Figure 6.2 is provided where a Consumer can keep changing the password for security reasons.

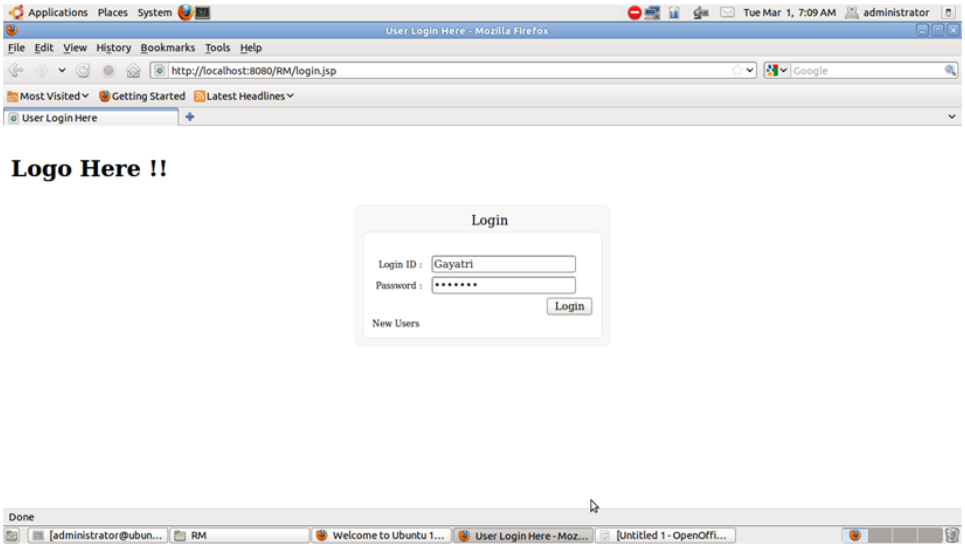

Figure 6.2: Login Screen.

The Interface depicted in Figure 6.4 displays the different registered Consumers to the Administrator.

The Resources can be added by the authorized users through the interface depicted in Figure 6.5 and 6.6.

An Interface that depicts the different shell files to schedule the jobs of different types. Jobs in C,C++,Java can be submitted well in advance .Jobs are the programs that are submitted by the Consumers to be scheduled on the Eucalyptus Cloud.

An Interface to depict the entry for the Slot. The authorized user can only change the slot as shown in Figure 6.8.

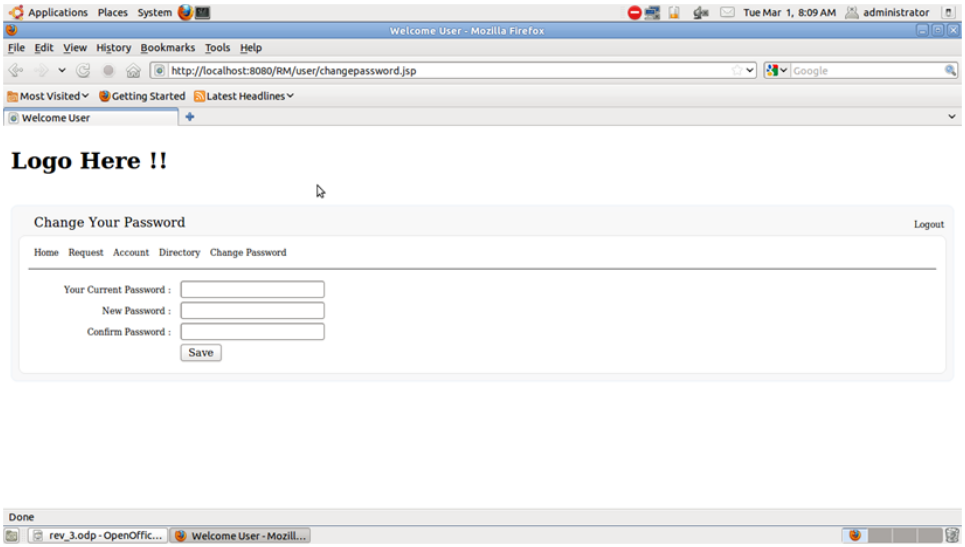

Figure 6.3: Change Password Option.

An Interface to select the start time and the end time for the Consumers for the job submission for execution as depicted in Figure 6.9.

An Interface to display the previous job submitted by the Consumer as depicted in Figure 6.10.

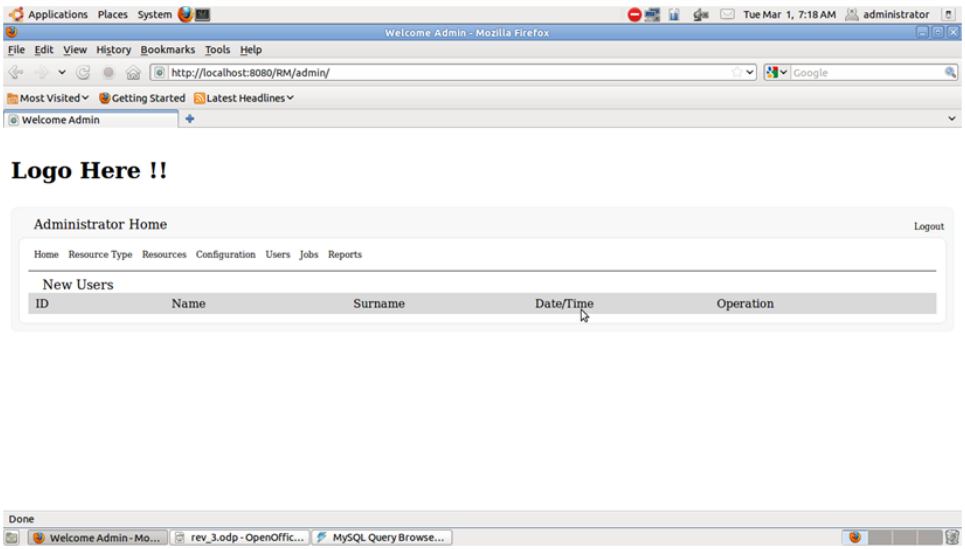

Figure 6.4: An Interface showing different Consumers to the Administrator.

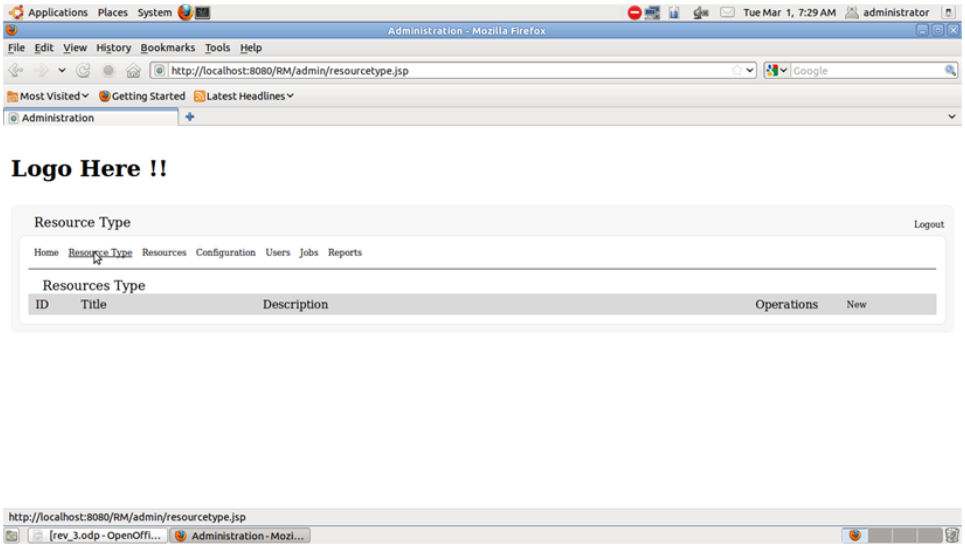

Figure 6.5: An Interface to add the types of Resources.

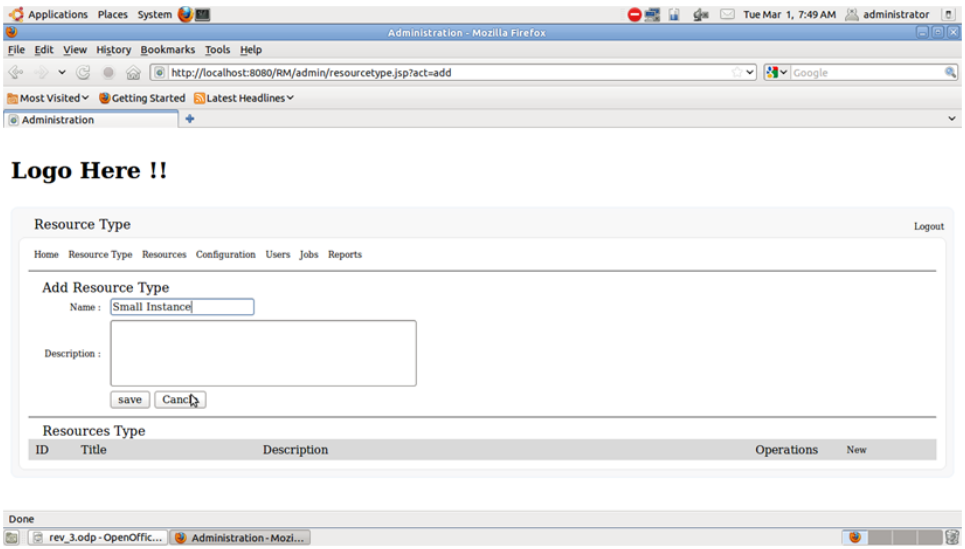

Figure 6.6: An Interface showing resource type as small instance.

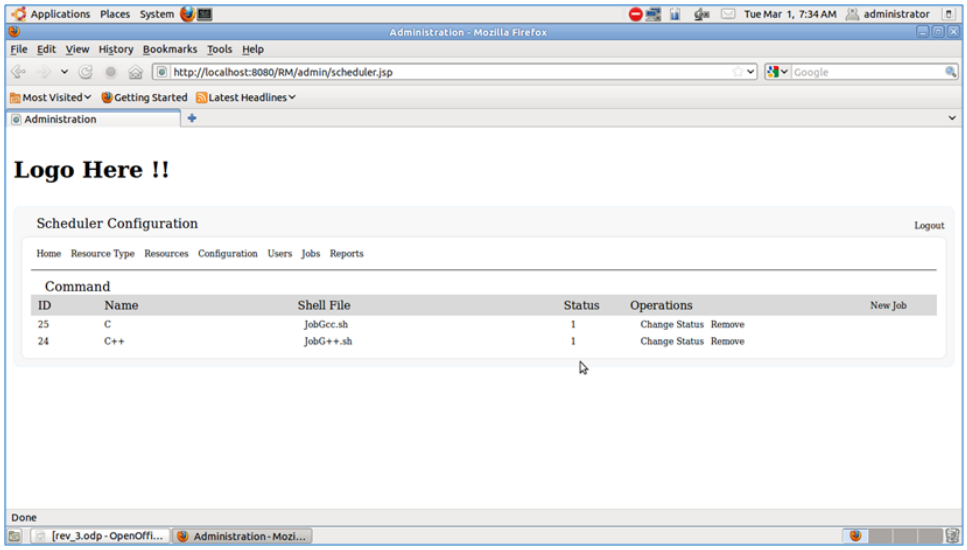

Figure 6.7: An Interface to display the different types of Shell files for different types of compilers to schedule the jobs

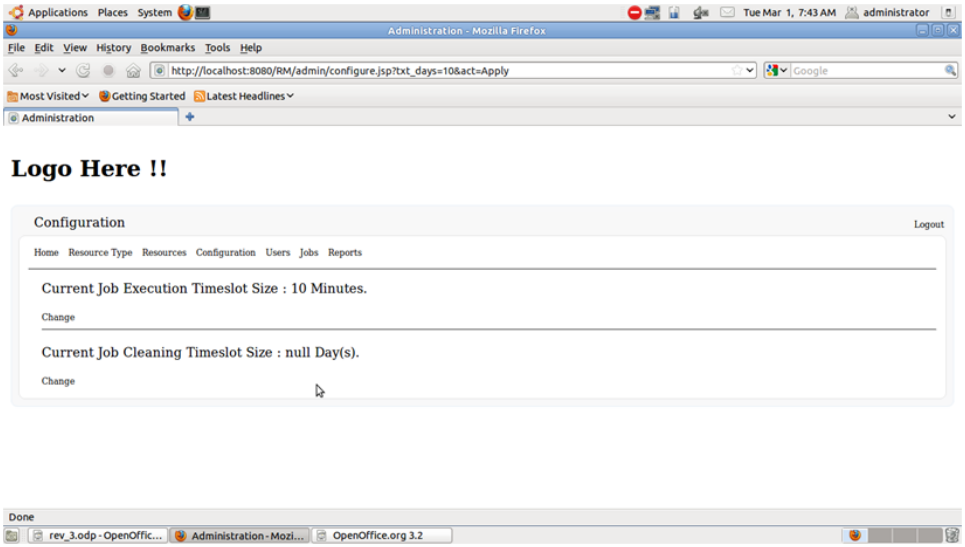

Figure 6.8: Slot Interface.

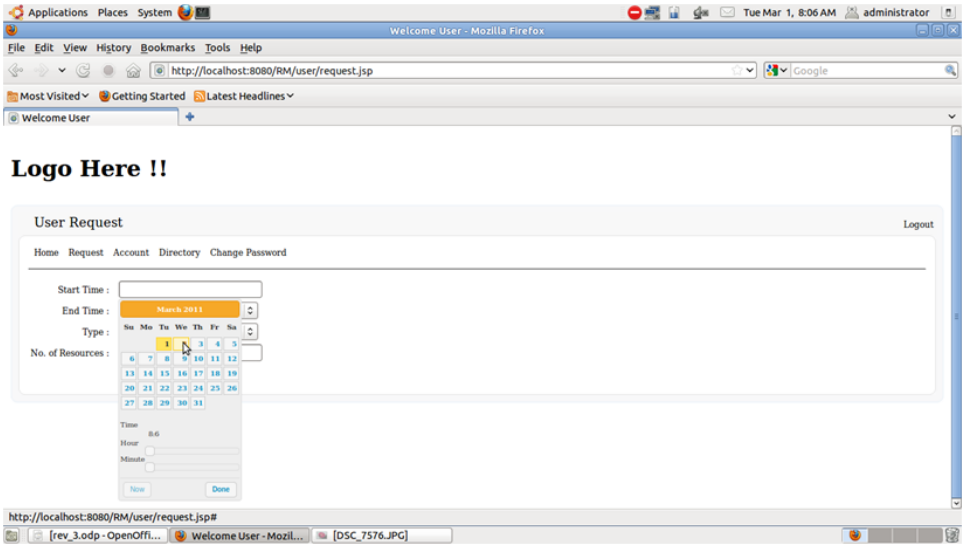

Figure 6.9: Slot Interface.

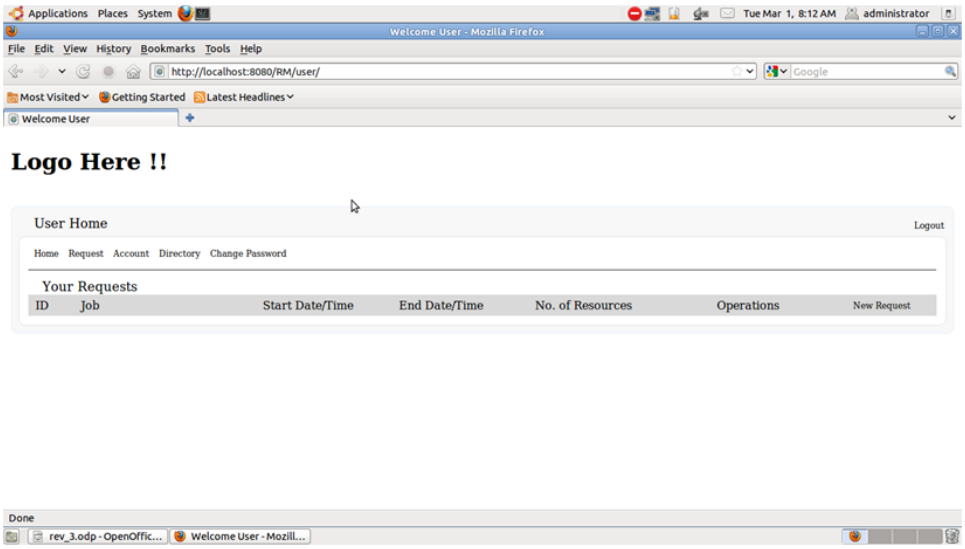

Figure 6.10: Previous Job Submitted.

Chapter 7

Performance Evaluation

This chapter focuses on the Performance Module of the Advance Reservation System a.k.a as Resource Management System. A functional structure of the Module is developed and highlighted in Figure : 7.1.

Eucalyptus Advance Reservation Queue (ECARQ) implementation includes a front end scheduling daemon a node scheduling daemon and a DbDaemon. Daemons are implemented using JSP,MySQL and Unix Shell Scripts that perform Job scheduling and execution. Web based GUI's are provided for all the major operations.

- User Interfaces : ECARQ provides several User Interfaces for submitting a Job Releasing a Job and querying about the job status and information. A User can submit several parallel Jobs using these User Interfaces.
- Job Scheduling :The Scheduling is performed with the aid of several other modules.When the module is started the resources are loaded from the predefined scripts which reads values from the eucalyp.conf which is a monolithic file used to control the Eucalyptus Clouds.
- Job Execution : ECARQ supports both batch Jobs and Interactive Jobs. It is guided by the scheduling Information and responsible for actual assigning /releasing node access to the corresponding user when a job is started / finished. It also release the resources for early completed jobs and terminates the uncompleted Jobs when scheduled time arrives.
- Plugin Scripts : ECARQ has to adapt different parallel program execution environments. This is implemented by writing specific shell scripts to interface the ECARQ job execution to different environments.

In order to evaluate the performance of our proposed data structure, i.e. ECARQ with Sorted Queue two nodes have been setup. The slot time can be set from the Interface.Only the Administrator or any authorized user has the right to set the various values for the slots. Various screen shots are available and are specified in Chapter 6. The parallel programs(jobs) which are submitted, compiled and executed are Matrix Multiplication and Bubble Sort.

The Algorithm Complexity is specified in Fig7.2

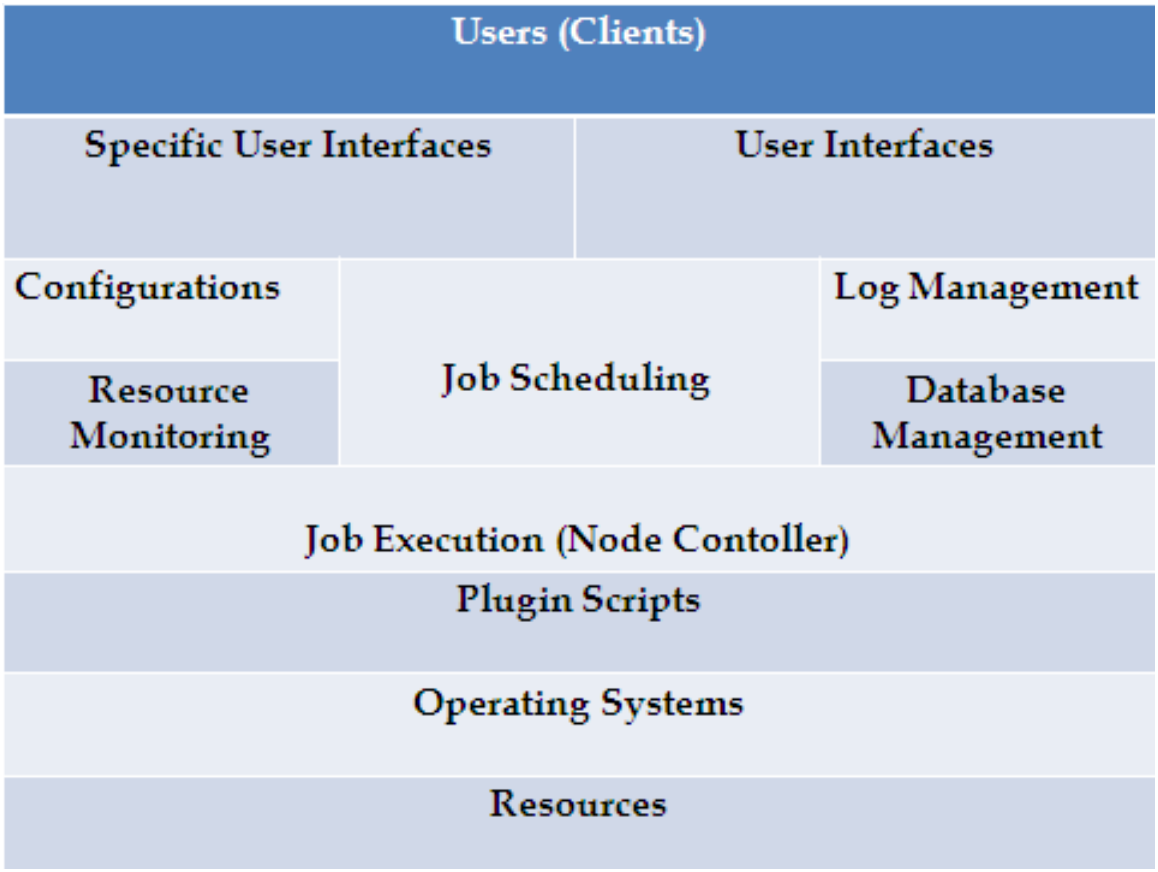

Figure 7.1: ECARQ Functional Structure.

The Figure 7.3 specifies that the System is tested with 5 Users and their respective Parallel Jobs.

The Figure 7.4 specifies the Accepted Jobs,Suggestions Given Errors and Other Details.

The Figure 7.5 suggest the Free Slot Available for the User. These suggestions are implicitly given when a Consumer selects a slot which is already booked by some other Consumer. A Consumer can thus select the suggested Interval and thus confirm the specified slot for himself/herself.

These suggestions are intelligent suggestions as it would display the Free slots to the Consumers, thus utilizing the resources efficiently and thus Resources managed effectively.As the free slot would otherwise go waste.

The Figure 7.6 focuses on the No of Jobs Rejected(canceled) by the system due unavailability of Services of Cloud. These jobs are scheduled later when the slots are Free.Now again the Free slots are utilized by the System very smartly.

Operations	Time Complexity
Add	$O(k+n_{buk})$
Delete	$O(n_{buk})$
Search	$O(n_{buk})$
Note : k is the number of reservations	
: n_{buk} is the no of buckets	

Figure 7.2: Complexity of the Algorithms.

7.1 Summary

This chapter focuses on ARM or RMS. Major srceens related to Performance Evaluation are highlighted here. The screens suggesting the user of suggestions if the slot is not available here are also discussed.These suggestions are termed by us as intelligent suggestions.

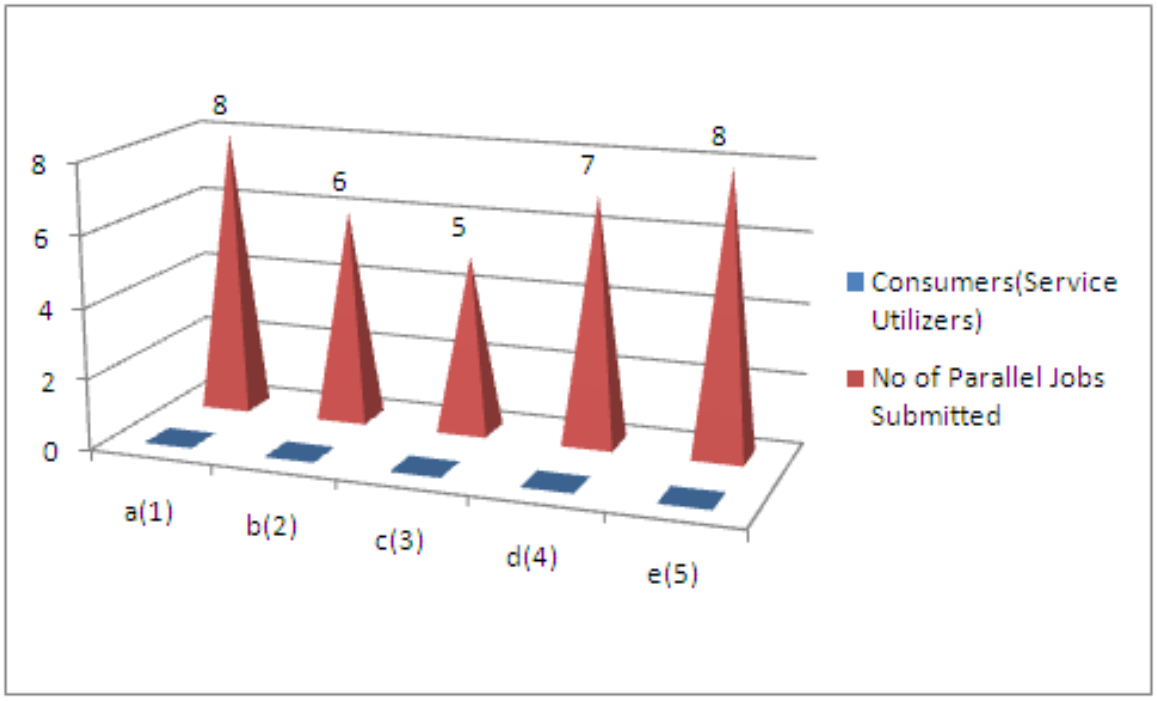

Figure 7.3: User and Parallel Job Submitted by the specified Users.

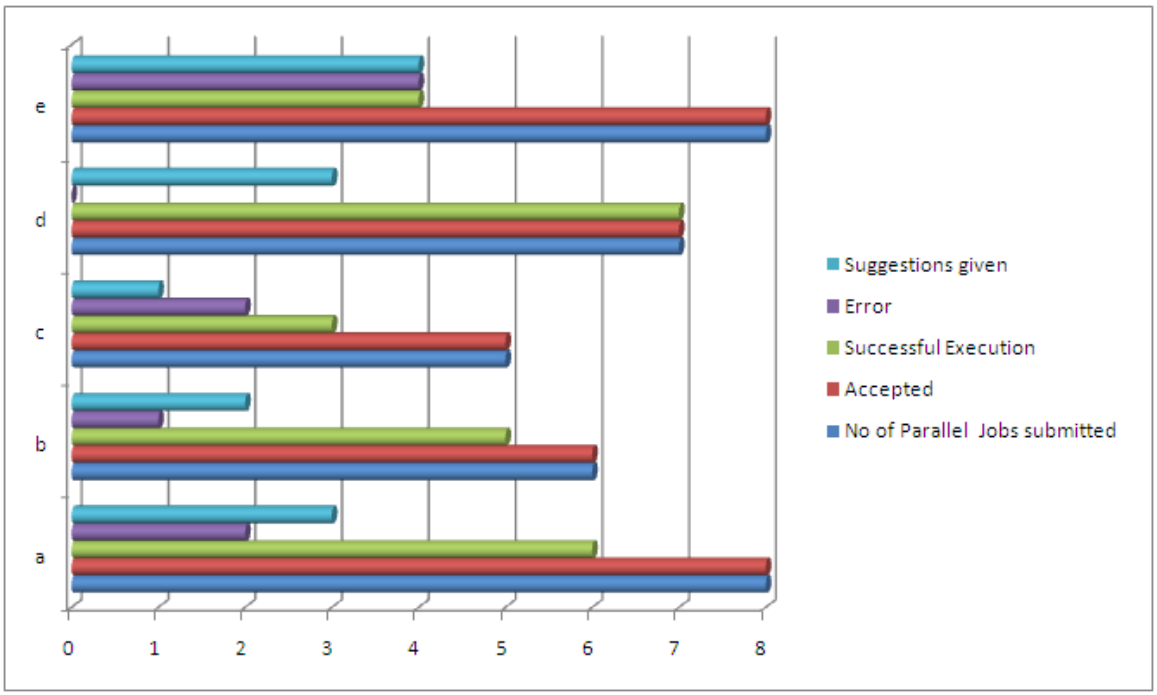

Figure 7.4: Job Accepted, Suggestion Given and Errors for Job Submitted .

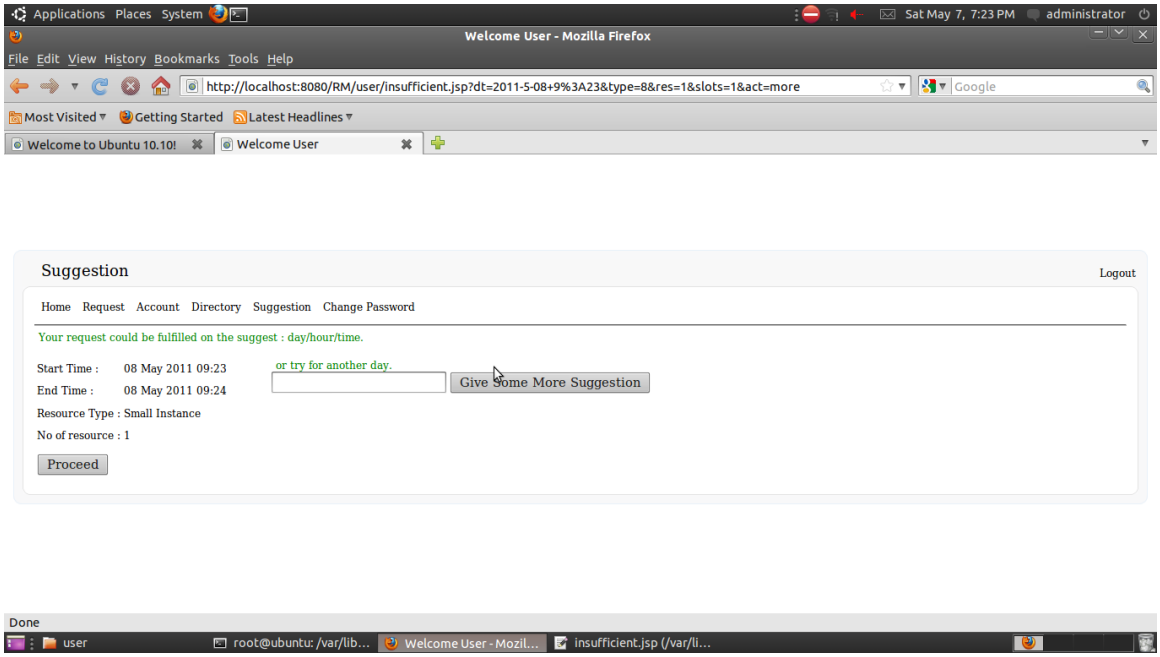

Figure 7.5: Suggestions.

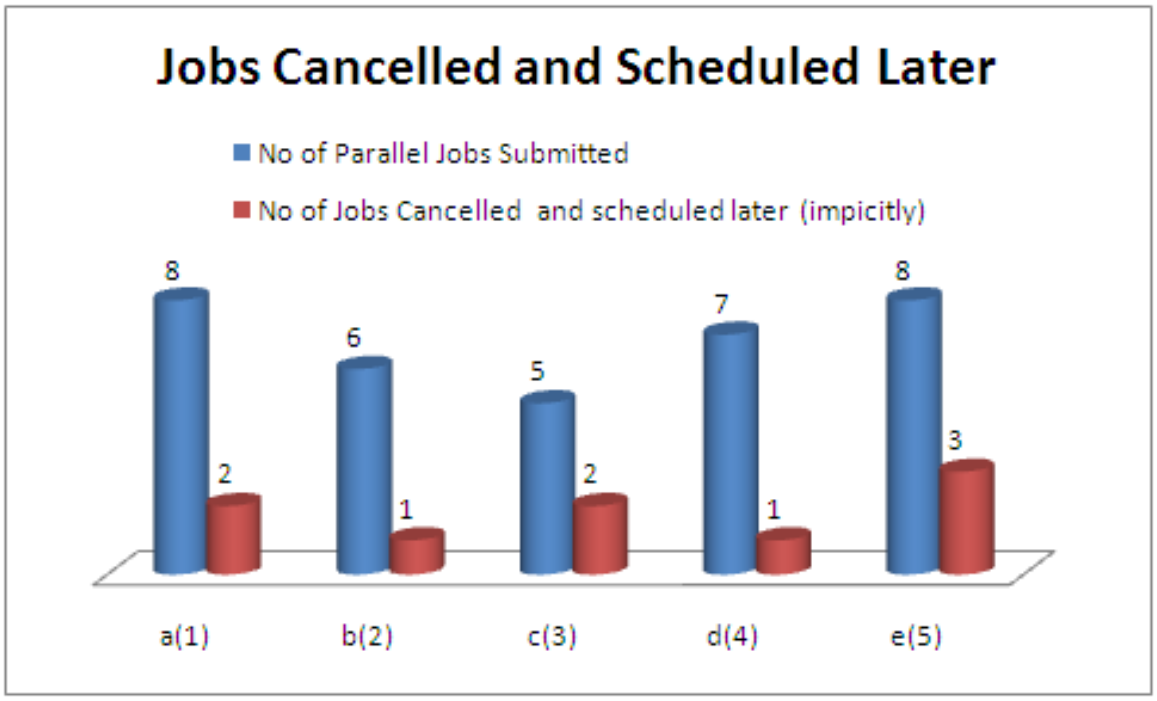

Figure 7.6: Jobs Canceled and Scheduled Later.

Chapter 8

Conclusion and Future Direction

Cloud technologies represent a significant achievement towards the aggregation of networked resources for solving large-scale data-intensive or compute-intensive applications.This thesis proposes the use of advance reservation to ensure the specified resources are available for applications when required. The studies here are carried out through the Eucalyptus Cloud . In this chapter, there are highlight on the thesis contributions and possible future directions.

AR is useful in scenarios where lease would turn out cheaper than installation of Resources. Common Resources which could be requested or leased are Nodes (VM), network bandwidth, memory and disk space. AR in a Cloud thus allows consumers to achieve synchronized access for their applications to be executed in parallel and assures the availability of requested resources to the consumer at specified future times. With AR thus resources could be managed effectively and would offer a better QoS.

8.1 Conclusion

This thesis describes the development of Advance Reservation Module for the Cloud An Eucalyptus Cloud has been built successfully on Ubuntu 10.10 (Maverick Meerkut). ARM Modules is built in JSP (Java Server Pages), Tomcat 6 and MySQL 5 on Ubuntu 10.10.

Free slots leads to Wastage of Resources.AR thus aids in better Utilization of Resources as these free slots could be utilized else which would otherwise go waste. AR is thus useful is scenarios where lease would turn out cheaper than installation of Resources. Common resources which could be requested or leased are Nodes (VM), network bandwidth, memory and disk space or a combination of these. AR in a Cloud thus allows consumers to achieve synchronized access for their applications to be executed in parallel and assures the availability of requested resources to the consumer at specified future times. With AR thus resources could be managed effectively and would offer a better QoS.

This thesis provides a case for an elastic reservation model, where users can self-select or choose the best option in reserving their jobs, according to their QoS needs and availability of resources.

In this model, the Cloud system has a Reservation System (RS) and a DbDaemon. The RS is responsible for handling reservation queries and requests. When the RS receives a reservation query or request, it searches for availability. More specifically, the RS communicates with the DbDaemon, for this request. Therefore, the primary role of the

DbDaemon is to store and update information about resource availability as the time progresses.

A well-designed data structure provides the flexibility and easiness in implementing various algorithms. This thesis suggests a data structure for administering reservations efficiently in the DbDaemon. The new data structure is called Eucalyptus Advanced Reservation Queue (ECARQ).

ECARQ has the following advantages:

- a fast O(1) access to a particular slot.
- able to reuse these slots for the next time interval, assuming that the length of a reservation is less than 30 days; and
- built only once in the beginning

The algorithms which are used here consider the duration and number of required compute nodes as soft constraints for a given reservation query. Thus, it aims to find a solution or alternative offers within the given time interval for clients to choose themselves. In addition, the algorithm aims to reduce fragmentation or idle time gaps caused by having reservations in the system.

Having a degree of flexibility in the reservation requests allows an improvement in the resource utilization.Advance Reservation thus aids in better Utilization of Resources as Free slots leads to Wastage of Resources, Free slots could be utilized in a better manner.

8.2 Future Direction

This thesis suggests several future directions to further enhance AR for Clouds. The future directions are related to the three key functionalities of Clouds, i.e. job scheduling, resource management and data management.

8.2.1 Incorporating Resource Failure Model

The Resource Scheduler presented in this thesis assume that all the compute nodes are available for execution. However, in reality, some of these nodes may not be available at some point in the future due to maintenance or upgrade (e.g. software, hardware and security). Thus, the Resource Scheduler needs to consider a case where several nodes fail during execution. The addition of a resource failure model to the job scheduling problem will present another challenge to the Resource Scheduler and DbDaemon. The Resource Scheduler needs to interact with the RMS and the DbDaemon to find suitable solutions. Such solutions can be migrating the affected jobs to other available nodes either located internally or externally, postponing these jobs to later times, or providing them with some compensation costs. However, these decisions needed to be chosen carefully as they may reduce the overall resource revenue and disrupt other reservations and existing jobs in the queues. As such, incorporating the resource failure model provides an interesting and exciting research problem.

8.2.2 Integrating Various Types of Resources

As mentioned previously, common resources that can be reserved are compute nodes (CNs),storage elements (SEs), network bandwidth or a combination of any of those.

However, this thesis is mainly focusing on reserving compute nodes. Therefore, allowing users to reserve a combination of resource types is highly desirable, since various applications, especially in the area of Cloud, can be modeled and studied.

Appendix A

List of Publications

My paper entitled "Effective Resource Management in Clouds Using Advance Reservation" has been published in 2010 International Conference on Intelligent Network and Computing (ICINC 2010)held at Kuala Lumpur Malaysia in November 26 - 28 2010, with IEEE Catlog Number : CFP1076K-PRT and ISBN 978-1-4244-8270-2 with page nos 380 to 383.

Bibliography

[1] Rajkumar Buyya1,2, Chee Shin Yeo1, and Srikumar Venugopal and Grid Computing and Distributed Systems (GRIDS) Laboratory Department of Computer Science and Software Engineering. Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities. In the Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications: page 5 - 13

[2] Borja Sotomayor ,Ruben Santiago Montero Ignacio Mart?n Llorente, Ian Foster. Resource Leasing and the Art of Suspending Virtual Machines, In the Proceedings of 2009 11th IEEE International Conference on High Performance Computing and Communications: page 59 - 68.

[3] N. Fallenbeck, H.-J. Picht, M. Smith, and B. Freisleben. Xen and the art of cluster scheduling. In VTDC '06: Proceedings of the 1st International Workshop on Virtualization Technology in Distributed Computing. IEEE Computer Society, 2006.

[4] W. Emeneker and D. Stanzione. Efficient Virtual Machine Caching in Dynamic Virtual Clusters. In SRMPDS Workshop, ICAPDS 2007 Conference, December 2007

[5] N. Kiyanclar, G. A. Koenig, and W. Yurcik. Maestro VC: A paravirtualized execution environment for secure on-demand cluster computing. In CCGRID '06: Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGRID' 06), page 28. IEEE Computer Society, 2006.

[6] J. P. Walters, B. Bantwal, and V. Chaudhary. Enabling interactive jobs in virtualized data centers. In Cloud Computing and Applications 2008 (CCA08), 2008.

[7] Anthony Sulistio and Rajkumar Buyya : A GRID SIMULATION INFRASTRUCTURE SUPPORTING ADVANCE RESERVATION

[8] Borja Sotomayor, Ruben S. Montero, Ignacio M. Llorente, and Ian Foster: An Open Source Solution for Virtual Infrastructure Management in Private and Hybrid Clouds, IEEE INTERNET COMPUTING, SPECIAL ISSUE ON CLOUD COMPUTING July 2009

[9] Alain Roy Scheduling Working Group University of Wisconsin-Madison,Forschungszentrum Jlich GmbH May 2002

[10] Xindong YOU, Xianghua XU, Jian Wan, Dongjin YU School of Computer Science and Technology Hangzhou Dianzi University Hangzhou, China,youxindong@hdu.edu.cn, wanjian@hdu.edu.cn: RAS-M:Resource Allocation Strategy based on Market Mechanism in Cloud Computing 2009 IEEE page 256 - 253

[11] Scheduling with Advanced Reservations Warren Smithy Ian Foster Valerie Taylory Mathematics and Computer Science Division Argonne National Laboratory, Argonne, IL 60439 fwsmith,fosterg@mcs.anl.gov

[12] Lizhe Wang, Jie Tao, Marcel Kunze Institute for Scientific Computing, Research Center Karlsruhe Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen, Germany Alvaro Canales Castellanos, David Kramer, Wolfgang Karl Department of Computer Science, University Karlsruhe (TH) 76128 Karlsruhe, Germany : Scientific Cloud Computing: Early Definition and Experience ,In the proceedings of the 10th IEEE International Conference on High Performance Computing and Communications, page 825 - 830

[13] Borja Sotomayor, Kate Keahey, Ian Foster : Combining Batch Execution and Leasing Using Virtual Machines, HPDC'08, June 23-27, 2008, Boston, Massachusetts, USA, ACM.

[14] Luqun Li An Optimistic Differentiated Service Job Scheduling System for Cloud Computing Service Users and Providers: 2009 Third International Conference on Multimedia and Ubiquitous Engineering. page 295 - 299

[15] Massimiliano Rak , Emilio P. Mancini,UmbertoVillano PerfCloud: GRID Services for Performance-oriented Development of Cloud Computing Applications, In 2009 18th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises ,page 201 - 206

[16] Eddy Caron, Frederic Desprez, David Loureiro , Adrian Muresan Cloud Computing Resource Management through a Grid Middleware: A Case Study with DIET and Eucalyptus ,In the proceedings of 2009 IEEE International Conference on Cloud Computing, page 151 - 154

[17] Junjie Peng School of computer science and High performance computing center Shanghai University Shanghai,China ,Xuejun Zhang Qinghua machinery factory at Changzhi Changzhi, China,Zhou Lei, Bofeng Zhang, Wu Zhang, Qing Li School of computer science and High performance computing center Shanghai University Shanghai,China: Comparison of Several Cloud Computing Platforms, In the proceedings of the Second International Symposium on Information Science and Engineering

[18] http://open.eucalytus.com/

[19] http://maven.apache.org/

[20] http://linuxers.org/article/kickass-features-that-will-make-ubuntu-1010-the-perfect-10

[21] http://www.ubuntu.com/cloud/why-ubuntu

[22] http://manpages.ubuntu.com/manpages/maverick/man5/eucalyptus.conf.5.html

[23] https://help.ubuntu.com/community/UEC/CDInstall

[24] http://fnords.wordpress.com/2010/02/21/ubuntu-enterprise-cloud-autoregistration-features/

[25] http://en.wikibooks.org/wiki/LaTeX/DocumentStructure

[26] http://en.wikibooks.org/wiki/LaTeX/AlgorithmsandPseudocode

Index