3-2015

Mobile Agent Based Cloud Computing

Vijaya Rauniyar
St. Cloud State University

MOBILE AGENT BASED CLOUD COMPUTING

By

Vijaya Rauniyar

B.E., Tribhuvan University, Nepal, 2010

A Starred Paper

Submitted to the Graduate Faculty

of

St. Cloud State University

in Partial Fulfillment of the Requirements

for the Degree

Master of Science

St. Cloud, Minnesota

March, 2015

This starred paper submitted by Vijaya Rauniyar in partial fulfillment of the requirements for the Degree of Master of Science at St. Cloud State University is hereby approved by the final evaluation committee.

Donald Hamer
Chairperson

[illegible]

Dennis Guster

Patricia Hughes
Dean
School of Graduate Studies

ABSTRACT

Cloud Computing is becoming a revolutionizing computing paradigm. It offers various types of services and applications that are being delivered in the internet cloud. The services aim at providing reliable, fault tolerant dynamic computing environment to the user and offers computing resources as per demand. Skype, Dropbox, and Yahoo mail are some of the cloud services that have major impact in our lives. Several measures are taken to maintain the quality of its service in the cloud and to make IT infrastructure available with low cost. This paper presents various aspects of Cloud Computing, its implementation features, challenges and also explores the potential scope for research. The major section of this paper includes surveys of studies related to the possibilities of integrating Mobile Agents in Cloud Computing, since these technologies appear to be promising and marketable. Thus, the paper focuses on resolving challenges and bolstering services of Cloud Computing by utilizing Mobile Agent technology in various aspects of Cloud Computing.

ACKNOWLEDGEMENT

I would like to express my special gratitude and thanks to my advisor Professor Dr. Donald Hamnes. He has spent countless hour proof-reading my writings and directed my effort in the right direction. His guidance, continuous supervision and encouraging approach has made it possible to move my survey idea to a completed paper. He has been a tremendous guru to me. I would also like to express my sincere appreciation to my committee members, Dr. Jie H. Meichsner and Dr.Dennis Guster, for serving as my committee members in spite of their busy schedule. Thank you professors for allowing me an opportunity to share my work with you, and for providing excellent comments and suggestions in my defense. In addition, thanks to St. Cloud State University: Department of Computer Science and Information Technology, Learning Resource Services, School of Graduate Studies, Writing Center and all faculty and staff who helped me by providing a learning environment, necessary research materials and guidance whenever needed. Their assistance has made a huge contribution during the completion of my work.

This Starred Paper is dedicated to my parents, Mr. Bindeshwar Prasad Gupta and Mrs. Sudha Gupta, for believing that I could do it; for giving me an opportunity to get my higher degree from one of the best institutions in the United States and for being helpful throughout my life. This paper is also dedicated to my best friend, Sudip Pariyar, who has always been there for me and for providing me a push needed to complete my work.

TABLE OF CONTENTS

LIST OF TABLE

LIST OF FIGURES

Chapter 1

INTRODUCTION

CLOUD COMPUTING

Cloud Computing is an internet based computing system. It not only refers to software services delivered over the internet, but also all those services that are provided by the hardware and systems software in the datacenters [1]. This is what is termed a 'Cloud'. It has the ability to bring a revolutionary change in the field of the IT industry.

Cloud Computing has brought a remarkable difference in the field of both IT hardware and software. Cloud Computing is advantageous not only in modeling the design and purchase of IT hardware, but also has provided an efficient use of software with more flexibility as a service [1]. It has central remote servers which assist in data and applications storage. The consumers and businesses can use applications via cloud without even installing it and can also access their personal files from anywhere and whenever required [2]. The IT industry no longer has to spend its large capital outlay on hardware infrastructure in order to deliver any innovative ideas from developers and its services to end users [1].

In other words, Cloud Computing can be referred to as the next generation of computation. People can look for and can access anything in the Cloud. People do not need to be proficient with, or have control over the technology infrastructure which is supporting the service [3]. Some of the services of Cloud Computing include sharing of software and platforms over the internet, utility computing, virtualization, etc. [4]. Several companies like Google, Amazon, and Microsoft are providers of Cloud Computing services [5].

Characteristics

Centralized data storage, greater processing capabilities and bandwidth facilitate the Cloud Computing technology for enhanced and efficient computing [2]. Cloud Computing has several characteristics like cost-effectiveness, reliability, security, easy maintenance etc. [6] However, the major ones are explained below [6]:

Shared infrastructure. Use of virtualized software model in the Cloud, facilitates the sharing of physical services, storage devices and network capabilities and hence enhances the utilization of cloud infrastructure across users.

Dynamic provisioning. According to the current demand requirements of the users, services in the Cloud are delivered. Software automation technique facilitates the process of utilization by expansion and contraction of service capabilities as per requirement.

Network access. Cloud Computing deploys several services in the Cloud, from business applications to the latest applications on cellphones. Hence, all these services in the Cloud can be accessed from a wide range of devices such as PCs, laptops, smartphones, etc. across the internet.

Managed metering. In order to provide reporting and billing information to customers, Cloud Computing uses a metering system. This helps manage and optimize the service as well as figure out the amount of service used by the consumers.

Merits and Demerits

In spite of all these characteristics, there are several pros and cons of using Cloud Computing. Users have to shift all their data to the Cloud. So, there still lies a debate about whether the end users are willing to do that and quit using the services on their local machines and devices [3]. Some of the advantages and disadvantages of using Cloud Technology are listed below.

Merits. Benefits of using Cloud Computing are as follows [3]:

- Reduced Cost: Cloud users have to pay only for the amount of service needed and used. This helps in reducing unnecessary cost and saves an organization's money.
- Increased Storage: Cloud Computing provides a larger storage capacity than a person can store on a private computer system.
- Highly Automated: IT personnel no longer have to worry about updating the software as part of the maintenance activity. Cloud service provider is responsible for software update in cloud.
- Mobility: Information in the Cloud can be accessed from anywhere and whenever required.

- Allows IT to Shift Focus: It helps IT personnel to focus more on delivering other innovative ideas rather than dealing with computer issues that cloud service providers can handle.

Demerits. Some of the issues found while shifting to Cloud Computing are as follows [3]:

- Security: It is still a questionable topic to determine if there has been any standard security measures applied in the Cloud.
- Reliance on third party: Cloud users are still hesitant in trusting the provider in the cost of losing their control over their own data.
- Cost of Transition: Cost of shifting data from its existing data center to the Cloud architecture is sometimes impractical to users.
- Uncertainty of benefits: Users are still not convinced about the long term benefits that the Cloud can provide.

CLOUD ARCHITECTURE

In order to implement IT services without investment, organizations are offered a business model by Cloud Computing [7]. Two basic cloud models are:

- Service Models
- Deployment Models

Some of the features of these models are discussed below.

Service Models

Cloud Computing provides scalable IT-related services across the internet and to numerous consumers. It is broadly classified into three services: — “SaaS”, “PaaS” and “IaaS” as shown in Figure 1.

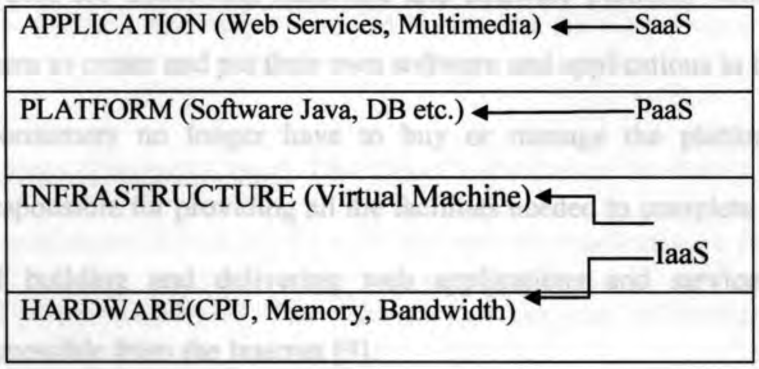
APPLICATION (Web Services, Multimedia)
SaaS
PLATFORM (Software Java, DB etc.)
PaaS
INFRASTRUCTURE (Virtual Machine)
IaaS
HARDWARE(CPU, Memory, Bandwidth)

Figure 1: Cloud Computing Architecture [adapted from 8]

Software as a Service (SaaS)

- SaaS is a software delivery model which is also referred to as “on-demand software” [9]. Consumers via web browser can access and use the desired software application service and associated data that are hosted in the Cloud centrally [8, 9]. For example [7] Gmail is a SaaS. Google is the provider, and the one who uses the service are consumers. Today SaaS is commonly used in various business applications like customer relationship management (CRM), management information systems (MIS), enterprise resource planning (ERP), invoicing, human resource management (HRM), content management (CM) and service desk management [9].

Platform as a Service (PaaS)

- In this model, a computing platform is provided by cloud providers. The platform may include various operating systems, software platforms, database and servers for supporting web services [10]. Consumers can access the underlying hardware and software platform which can assist them to create and put their own software and applications in the Cloud [6]. Consumers no longer have to buy or manage the platforms. PaaS is responsible for providing all the facilities needed to complete the life cycle of building and delivering web applications and services which are accessible from the Internet [9].
- Example of PaaS vendors are AWS Elastic Beanstalk, Cloud Foundry, Heroku, Google App Engine, Microsoft Azure, Amazon EC2, etc. [10].

Infrastructure as a Service (IaaS)

- Infrastructure as a Service is also called Hardware as a Service (HaaS). Here, consumers are provided with processing facility in terms of virtual machines, storage, and network connectivity. They can run operating systems but do not themselves control the Cloud infrastructure [6, 8]. Payment in IaaS is done on a per-use basis [9]. That means, consumers can simply go to one of the IaaS providers like Amazon Web Services EC2 (Elastic Compute Cloud), get their virtual server running in no time and pay only for the resources they use [8]. For example, Storage services are provided by AmazonS3, Amazon EBS and Computation services provided by AmazonEC2, Layered tech and so on [7].

Deployment Models

Below are some primary Cloud Computing deployment models each with specific characteristics that support service needs and the Cloud users in particular ways:

- Private (Internal) Cloud: The Cloud infrastructure is shared by a specific organization [8] and all the resources are deployed behind a firewall in case of private cloud [11]. The software and hardware infrastructure is owned by a user organization which is responsible for managing the Cloud and controlling the access to its resources [11]. It is built to provide services related to maintain security and privacy since the resources are not shared publicly with other organizations [6].
- Public (External) Cloud: Cloud infrastructure is publically accessible [7] on a commercial basis. Consumers can easily develop and deploy a service in the Cloud with very low cost [10]. Since services are available to the public, it might hamper effectiveness of this cloud since they don't have enough control over data, network and security settings [8].

Two additional types of cloud deployment models are defined by The National Institute of Standards and Technology (NIST) [12]. They are named as Community Cloud and Hybrid Cloud which are explained as follows.

- Community Cloud: It allows cloud infrastructure to be shared by multiple organizations with similar interests and requirements of a community [6, 8]. Since the cost of its establishment is shared among a number of organizations, it is beneficial in reducing the capital expenditure [10].
- Hybrid (Virtual private) Cloud: Cloud infrastructure is composed of a number of clouds of any type. It can be the combination of two or more clouds like private, public and community clouds. An organization in a hybrid cloud environment has an ability to provide and manage some information internally in an organization while some provided externally so as to deliver services in the Cloud [10].

For example, a company may archive its data to a publically accessible cloud service provider like Amazon Simple Storage Service (Amazon S3) and it may use an in-house storage for managing various customer operational resources. Data and applications can migrate from one cloud to another through their interfaces [6]. Compared to the public Cloud it provides higher security and control over data applications [8]. Also, it is more advantageous in terms of scalability and cost-effectiveness.

CLOUD COMPUTING CHALLENGES

Though Cloud Computing has several benefits like reliability, scalability, mobile accessibility, cost savings and many more, it still faces many obstacles in its efficient, fast, and secure delivery of services. Some of the challenges of Cloud Computing are as follows [13]:

- Dynamic Scalability: Based on the response time of user's queries entered in various applications, the compute nodes tend to dynamically vary in number. Hence, minimizing the scheduling delays is the major challenge for an effective and dynamic load management system in the Cloud.
- Multi-tenancy: An increase in the number of applications on the same compute node can result in a reduction in the amount of bandwidth allocated to each application. This may degrade the system's performance.
- Querying and Access: Easy querying for *Provenance Information* and also making the information securely accessible are also one of the major challenges for cloud environment [13]. *Provenance Information* refers to those resources that are persistent facts based on some particular source or URL [14]. The information does not change over time. For example, information presented on the life of dinosaurs based on evidences obtained from past research activities or any URL describing resources related to dinosaurs.
- Standardization: Since every organization has their own standards and use of APIs and protocols, integration and interoperability of all the services and applications become difficult to maintain.

- Reliability and fault-tolerance: In order to develop a reliable system, tools to test applications against fault tolerance and compute failures becomes mandatory.
- Debugging: Parallel and remote debugging is difficult in Cloud Computing.
- Security and Privacy: User is unaware of data storage location and who else can use it in the Cloud [13]. Deciding on the right cloud broker is the major challenge for maintaining the security mechanisms in the Cloud [8].
- Power: An enormous amount of power is consumed while offering services to meet the demands of users. An autonomic energy resource management system is one of the major requirements.

MOBILE AGENTS IN CLOUD COMPUTING

An agent is defined as a computer system that can independently decide for prospective actions needed in order to fulfill its design objectives [15]. A type of such agent is a Mobile Agent. An entity, which can migrate from one place to another in order to carry out its specific responsibility, is termed a Mobile Agent [16]. Mobile Agents migrate over the network from one device to another, without any loss of their codes or their states [17]. Since a Mobile Agent is known for intelligence and mobility, it plays an important role in various aspects of Cloud Computing. Since agents are sent to the tasks when needed that makes the exchange of messages local, this, in turn, assists in reducing all network loads [17]. It can also reduce communication costs by providing a means for the transportation of resources and services to the remote target environment [18]. Hence, it is widely used in applications distributed over wide area (slow) networks.

Cloud Computing plays a vital role in the evolution of on-demand information technology services and products [3]. Mobile Agents when combined with the Cloud environment accelerates its services even more and make Cloud Computing service more effective among users.

Some of the reasons for combining Mobile Agents with Cloud Computing [19]:

- Unlimited scalability: Mobile Agents make optimal use of resources in the Cloud. Whenever there is a demand for more resources, a transport agent is created that searches for available resources.
- More intelligence: Mobile Agents can be autonomous and auto-adaptable. These features assist the system with more intelligence in the discovery operation.
- Reduced consumption of band-width: Cloud Computing technology is always in need of greater bandwidth for better performance. Hence, Mobile Agents help in optimizing resources and facilitating the demand of bandwidth.
- Robust and fault tolerant: The ability of Mobile Agents to react dynamically to unfavorable situations makes the distributed system more robust and fault tolerant.

- Adapted to heterogeneous networks: Mobile Agents can easily adapt on networks based on heterogeneous hardware and software systems when it is programmed in the platform independent language JAVA.

Chapter 2

OBJECTIVES

Cloud Computing technology consists of large scaled computing infrastructure. Cloud infrastructure (hardware, software, services) frequently acts on different applications and service models having varying load, power consumption, and system sizes. Hence, a faster more reliable and efficient system must be utilized in order to provide an effective service in Cloud Computing technology. The main objective of this paper is to survey various features of Mobile Agent technology and to analyze its effectiveness in several areas of Cloud Computing. This paper aims at reviewing recent advances of Cloud Computing using Mobile Agent technology and discusses several areas of its applicability.

Since Cloud Computing dynamically reacts to adverse conditions, it is very important to design and employ techniques that can adapt in such situations. In many Cloud Computing architectures, cloud service providers are not able to provide an autonomous, auto-adaptable system which can assist in such environment. So, this paper intends to survey the effectiveness of mobile agents in a cloud environment.

This paper is presented with an aim of discussing an efficient use of Mobile Agent in Cloud Computing. It discusses related issues in order to provide a faster and efficient mechanism in selecting web service in the Cloud. It also describes techniques to maintain better data migration facilities in hybrid cloud. Besides these, it helps to understand monitoring facility of entire cloud system that provides a secure and reliable system.

Hence, it has illuminated several areas of Cloud Computing which could explain such techniques and methodologies well. As this paper will consolidate various areas of Cloud Computing interacting with Mobile Agents, it may also be useful to other students trying to survey in similar areas of Cloud Computing.

This paper is organized as follows: in the next chapter a brief comparison of cluster, grid and Cloud Computing is shown. Thereafter, a discussion of a cloud architecture, 'Cloud Agency' and implementation of Cloud Computing approach based on Mobile Agents is presented. Various interactions of Mobile Agents to Cloud Computing environment is detailed in the fifth chapter. This chapter includes a discussion on agent based service discovery, service negotiation and service composition. In addition to this, it also discusses cloud service migration, cloud scalability and maintaining trust in the Cloud based on the Mobile Agents. Finally, Chapter 6 concludes the research and presents the perspectives considered.

Chapter 3

COMPARISON OF CLUSTER, GRID AND CLOUD COMPUTING

Before discussing various ways by which agents interact with Cloud Computing let's first learn the difference between Cluster, Grid and Cloud Computing and the challenges they face.

Cluster Computing [13] is a collection of stand-alone parallel or distributed computers which are interconnected. High speed gigabit Ethernet, Myrinet, SCI or Infiniband connects all those clusters together. High availability, load-balancing and computing are the major areas where clusters are applied. Clusters share a single workload (e.g., computation) distributed among several computers. This is like a single virtual computer when several devices are connected together. In case of failure of system components, a cluster makes use of redundant nodes and improves the performance of the system.

Grid Computing [13] is generally used to resolve a single task. Computers from multiple administrative domains are combined together to reach its goal, and may then disappear just as fast. Grid Computing involves the aggregation of large-scale cluster computing based systems. When compared to Cluster Computing, the major difference is that grids are heterogeneous while a cluster is homogeneous. That means, the computers in the grid can have varying hardware and can perform operations on different operating systems while all computers in clusters have the same OS and hardware [20].

Grid and Cluster Computing have pre-reserved resources while Cloud Computing are demand driven, i.e. Resources are provided based on the needs of the consumers. The use of service can be easily measured in Cloud and Grid Computing while not in the case of Cluster Computing [21]. It is often discussed that the evolution of Cloud Computing occurred from Grid Computing and Cloud Computing relies on Grid Computing for its backbone and infrastructure support [22].

When cluster, grid and Cloud Computing are compared in detail, it is seen that grid and Cloud Computing are more convincing models in terms of standardizing APIs, security, interoperability, new business models, and dynamic pricing systems for complex services [13].

A comparison is represented in tabular form below:

Table 1

Features Differentiating Cluster, Grid and Cloud Computing [adapted from 13 & 21]

	Cluster	Grid	Cloud
Allocation	Centralized	Decentralized	Both
Resource Handling	Centralized	Decentralized	Both
Reliability	No	Half	Full
Security	Yes	Half	No
Interoperability	Yes	Yes	Half
Business Model	No	No	Yes
Rapid Elasticity	No	No	Yes
Scalable	No	Half	Yes
Resource pooling	Yes	Yes	Yes
Broad network access	Yes	Yes	Yes

Some of the major features that differentiates between cluster and grid with Cloud Computing are: use of resources, i.e., pre-reserved resources are generally available in case of cluster and grid environments while resources are available as per the demand and need of consumers in Cloud Computing.

Cloud can provide various tools that are applicable for testing for fault tolerance and several computing failures of applications executing in cloud [13]. This ensures reliability of the applications in Cloud Computing. Though providing a reliable system is a major challenge that Cloud Computing still faces, it is observed that Clouds provide a fully reliable system as compared to Grids and Clusters. Grids are half as much reliable as Clouds.

In Cloud, [13] users are unaware of the fact where their data is saved and who has access to it. Since there are many hackers around, users have no knowledge who can use their data. It is observed that Cloud Computing has minimal security as compared to Grid and Cluster Computing.

Rapid elasticity is most prevalent in Cloud Computing whereas a cluster or grid system does not support these features [21]. According to the National Institute of Standards and Technology (NIST), *elasticity* is the ability of the system to scale their resources up and down on demand basis. Consumers can purchase as much as they need, and for them Cloud Computing system appears to be infinite.

Another difference is that service usage in cluster environments are measured using some rudimentary metering functions while these features are very accurately measured in grid and Cloud Computing systems [21].

Interoperability [23] refers to the capacity of cloud to move workloads and data between clouds and from one cloud provider to another. According to various use cases discussed in [23], it is observed that syntactic interoperability (data exchange between systems) is present in the Cloud. However, cloud service cannot rely on semantic (meaningful data exchange) or organizational interoperability since the languages used by each provider are not fully understandable by another. Hence, data migration inside cloud still relies on documentation or agreements between providers. So, interoperability is half compared to cluster and grid in which programming like SOAP make data exchange more flexible. However, Cloud Computing solutions allow more flexibility to the system as consumers pay as per use.

Broad network access feature is available in all the technologies like cluster, grid and cloud as all of them host resources available in private cloud network to the wide range of devices like PCs, tablets and smartphones.

Cloud Computing can be summarized as a technology that mainly focuses on providing services such as software, platforms, etc., while clusters are mainly responsible for load balancing and high availability of a service. Grid Computing is a more promising model than cluster; however, compared to Cloud Computing, it usually has multiple ownerships in multiple locations which are connected to share combined resources while cloud is usually a single owner party.

Chapter 4

CLOUD ARCHITECTURE BASED ON MOBILE AGENTS

Cloud architecture described in [24] integrates Grid, Cloud and Mobile Agent technologies together. This Mobile Agent based cloud architecture, called *Cloud Agency*, is an integration of cloud on grid architecture. In Cloud Agency, Grid offers a high computation and secure platform infrastructure. Cloud provides a standard model and technologies which are suitable for distributing virtual clusters to final users, providing them with full administrative control. Mobile Agents assist the final users in configuring the available virtual clusters. The architecture developed is called 'Virtual Cluster Cloud Architecture' or 'Cloud Agency' in [24].

The architecture developed is divided into three components as follows [24]:

- <u>Services</u>: It provides the GRID environment with necessary functionalities. Grid Service termed as VCService (Virtual Cluster service) facilitates in the component implementation.
- <u>Images</u>: The Virtual Cluster Node images contain all the software necessary in integrating the VC (Virtual Cluster) into the GRID environment (a GT container) and the mobile agent platform (MAGDA).
- <u>Client</u>: It permits the end users of the system to cooperate in the Cloud environment.

Figure 2 shown below describes the overall architecture.

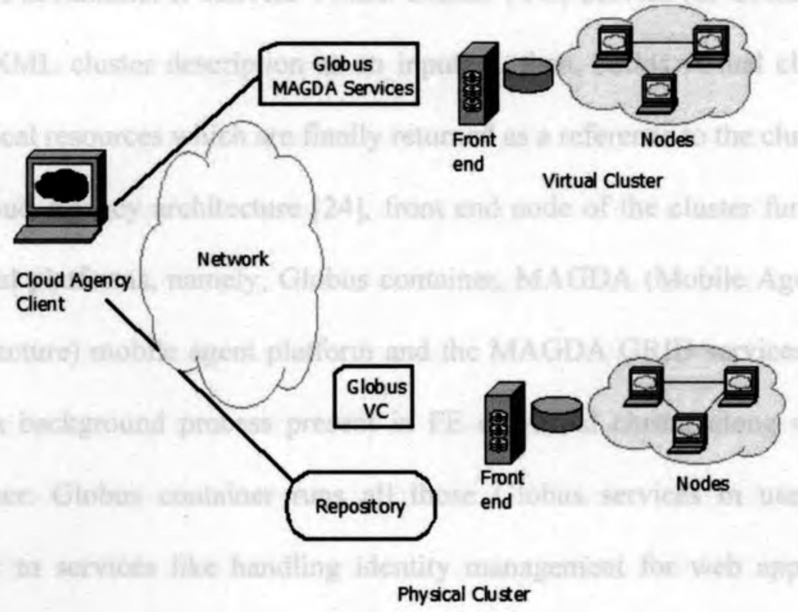

Figure 2: Architecture of Cloud Agency [adapted from 24]

Cloud Agency, a mobile agent based architecture, is developed with a goal of providing virtual clusters along with VC (Virtual Cluster) services to the underlying GRID. It offers solutions to cloud environment in order to perform various services with the help of virtual clusters. This platform also offers service called *Repository*, which maintains a collection of available agent based services in the system [24]. The application client is located in user's computer which has permission to access the GRID environment. Client cooperates with the system using the GRID services.

This model developed as shown in Figure 2 has a collection of clusters, each of which has front-end node (FE) and a set of computing nodes. Exchange of messages occurs among those computing nodes which act as a medium of communication on a private network. Cloud Agency [24] has an ability to add and configure services on virtual clusters at runtime. It delivers Virtual Cluster (VC) Service (or GRID service) that accepts XML cluster description as an input. It, then, builds virtual clusters on existing physical resources which are finally returned as a reference to the cluster.

In Cloud Agency architecture [24], front end node of the cluster functions in hosting several platforms, namely, Globus container, MAGDA (Mobile Agent based GRID Architecture) mobile agent platform and the MAGDA GRID services. Globus container is a background process present in FE of virtual cluster along with Jade Main container. Globus container runs all those Globus services in use. Globus services refer to services like handling identity management for web applications, providing a gateway to access computing infrastructure, etc. Configuring of the received virtual cluster occurs in the FE so as to host a Globus container with a valid certificate, the mobile agent platform and the MAGDA GRID services. Whenever VC starts, the preconfigured MAGDA services are made available to GRID infrastructure as resources which can be directly accessed by consumers.

The architecture [24] designed allows developing and deploying of agent based services in a Grid middleware. Mobile Agents play significant role in dynamically configuring the virtual cluster received from the Cloud, so that user can easily use it. Jade technology is used in the developed infrastructure which enables Mobile Agent execution. Jade signifies a FIPA (Foundation for Intelligent Physical Agents) following technology that makes use of a Mobile Agent platform. FIPA was founded by Swiss in 1996 which set standards for developing and setting computer software.

However, it dissolved in 2005 and was replaced by IEEE (Institute of Electrical and Electronics Engineers). Jade makes an environment available for agent's execution and also influences the agent's life cycle.

In order to configure virtual clusters dynamically [24], the Cloud architecture developed makes customized clusters available to users. The architecture also maintains a large repository of cluster virtual images. After the virtual cluster is configured using required software and refinements in operating system, it is delivered to the end user providing them with a full authority and control. However, without mobile agents, this architecture is not very efficient as it becomes difficult to add new functions to the virtual cluster and scale its configurations without altering the behavior of existing services. In Cloud Agency [24], Mobile Agent also plays an active role in providing solutions to several problems such as the following: by making the VC customization easier for the users so that user can easily choose among a set of existing services, and transfer them to the target VC.

Hence, Cloud Agency Dynamic Configuration model is developed [24]. A large number of agent-based services and a Repository (GRID) service together constitute the model. It assists users to retrieve all available services in order to deliver them to the target VC. The MAGDA-GRID integration, provides permission and control authority over platform service access.

The complete procedure of the working of this architecture is summarized in Figure 3 [24]:

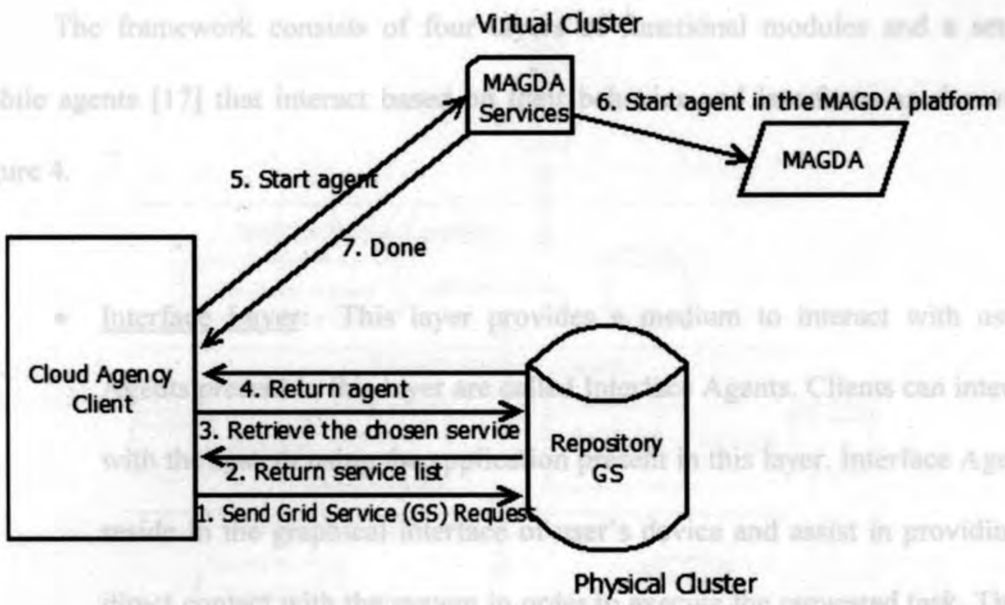

Figure 3: Summary of Cloud Agency Architecture [adapted from 24]

Another work presented in [17] presents an architecture which responds to user's needs for the execution of service in Cloud Computing. In this architecture, Mobile Agents play a very significant role. Mobile Agents improve the time required in running standard procedure on the server site and also can search for information in an improved way. For example, they can search for information based on the concepts, and can also correct queries based on the attached model. Agents are also able to generate their own knowledge bases from the data they obtain from each search. Sometimes, they utilize information that sites use to exchange data. Also, the searching process is greatly enhanced by agent's ability to communicate and cooperate among themselves.

The framework consists of four layers of functional modules and a set of mobile agents [17] that interact based on their behavior and interfaces as shown in Figure 4.

- Interface Layer: This layer provides a medium to interact with users. Agents present in this layer are called Interface Agents. Clients can interact with the system using the application present in this layer. Interface Agents reside in the graphical interface of user's device and assist in providing a direct contact with the system in order to execute the requested task. These agents capture all user's queries and send it to the suitable agents. It finally returns the result to the user.
- Mediation Layer: This layer creates mobile agents based on the requests made by Interface Agents. It contains Mediator Agents and Analyzer Agents. A Mediator Agent creates Mobile (Transfer) Agents for each user's request. These Mobile Agents migrate to task manager of the Cloud and search for the demanded service. Analyzer Agent, with the help of database present in the Mediation Layer, functions in selecting a list of cloud providers delivering the requested service. It then submits the result to the Mediator Agent.

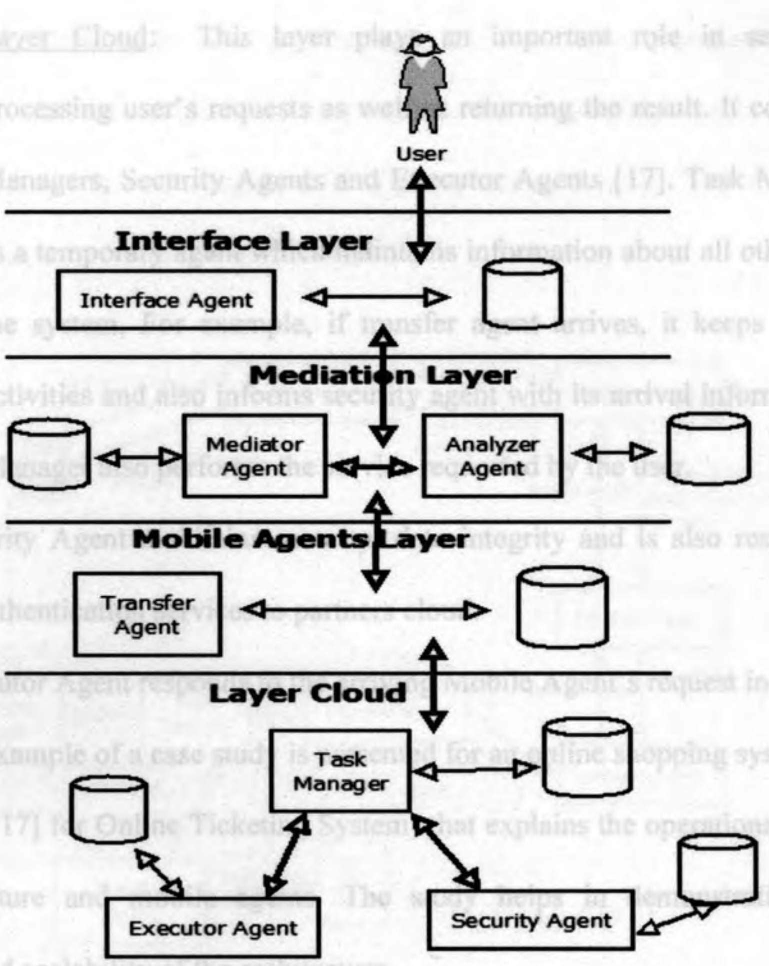

Figure 4: Layers of Mobile Agent Based Architecture [adapted from 17]

- Mobile Agents Layer: This layer activates a set of Transfer Agents generated by Mediator Agents to process each service requested [17]. Transfer Agents migrate to the specified cloud and search for a suitable service with the help of task manager present in that cloud. It then returns back to Mediator Agent to provide the result of the query made by Analyzer Agent. Finally, Transfer agent terminates itself.

- Layer Cloud: This layer plays an important role in servicing and processing user's requests as well as returning the result. It contains Task Managers, Security Agents and Executor Agents [17]. Task Manager acts as a temporary agent which maintains information about all other agents in the system. For example, if transfer agent arrives, it keeps track of its activities and also informs security agent with its arrival information. Task Manager also performs the service requested by the user.

Security Agent maintains security, data integrity and is also responsible for providing authentication services to partners cloud.

Executor Agent responds to the arriving Mobile Agent's request in the Cloud.

An example of a case study is presented for an online shopping system (similar example in [17] for Online Ticketing System) that explains the operational features of the architecture and mobile agents. The study helps in demonstrating validity, reliability and scalability of the architecture.

The study can be performed by simulating the activity in the Cloud for online shopping system. If a person wants to buy any product online through eBay or Amazon, there can be various criteria that should be taken into consideration. A product with similar features may vary in size, quantity, and manufacturer and subsequently their prices vary too. The purpose of this case study is to present an application for online shopping which helps user to choose a product with good price, size, color, the desired shipping time depending on each manufacturer available.

All the agents in this architecture use the Java language and Aglets development platform. Aglets are responsible for supporting agent development activity with its capacity to migrate it from one system to another. The steps followed by the agents are as shown in Figure 5 below.

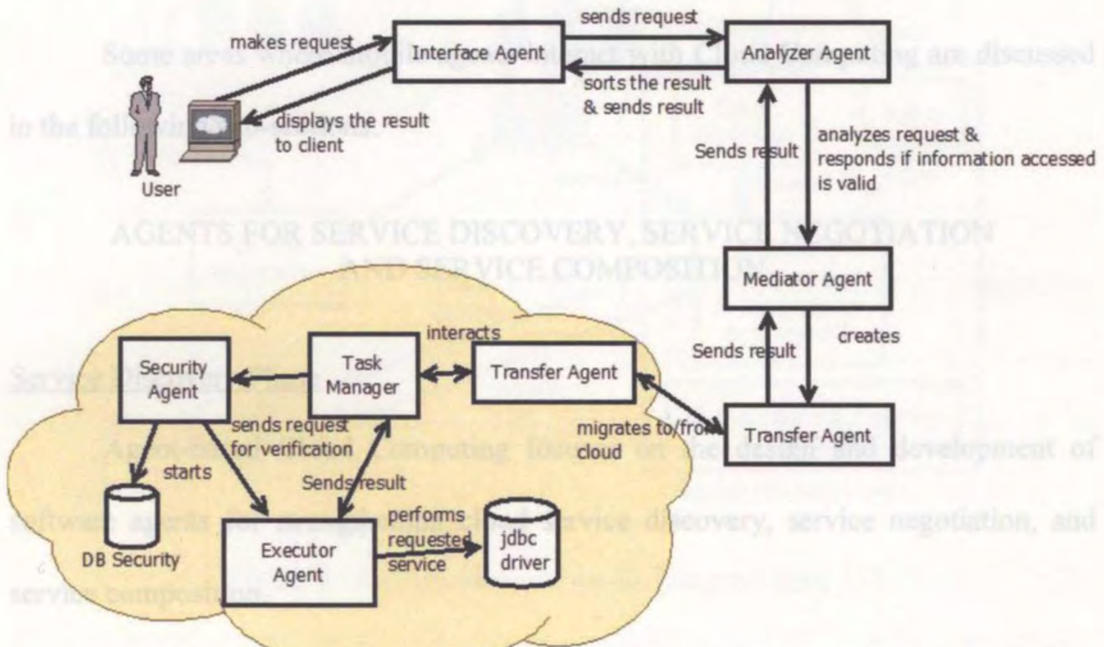

Figure 5: Architecture of Software [adopted from 17]

Chapter 5

INTERACTIONS OF MOBILE AGENTS WITH CLOUD COMPUTING

Some areas where mobile agents interact with Cloud Computing are discussed in the following sub-sections.

AGENTS FOR SERVICE DISCOVERY, SERVICE NEGOTIATION AND SERVICE COMPOSITION

Service Discovery Phase

Agent-based Cloud Computing focuses on the design and development of software agents for strengthening cloud service discovery, service negotiation, and service composition.

Searching of cloud services occurs in Service Discovery phase. The service discovery phase includes development of 'Cloudle' [25]. Cloudle is a search engine for cloud services that uses cloud ontology. Cloud ontology is a term used to represent a collection of cloud concepts and individuals of these cloud concepts. It also provides a basis to discover the relationship among those individuals [25]. In [15] Cloudle is developed as a tool for service discovery. An agent-based search engine, Cloudle functions in searching for cloud services for consumers that matches their functional, technical, and budgetary requirements [15].

Figure 6 below shows the architecture of Cloudle.

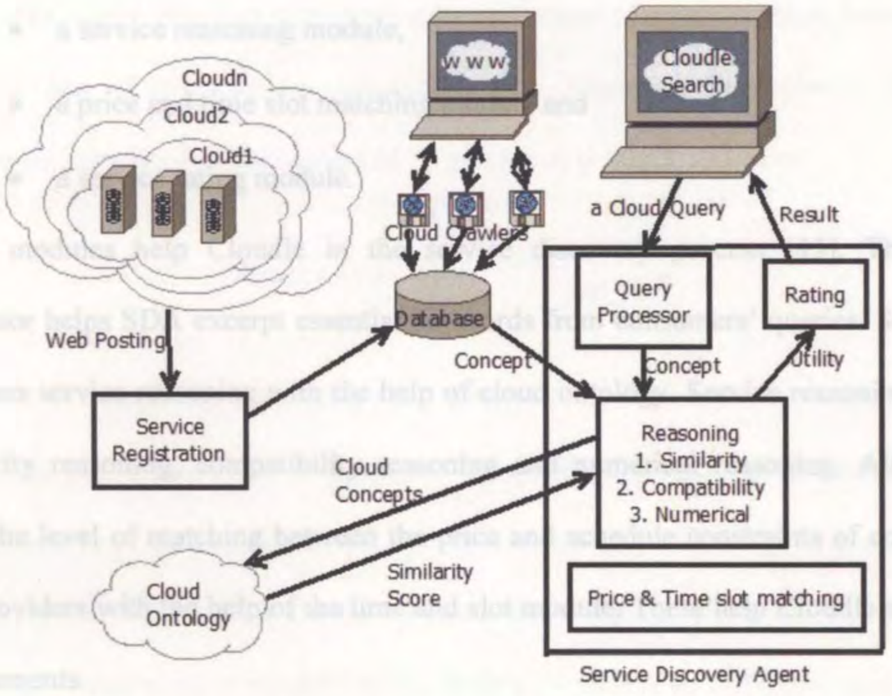

Figure 6: Architecture of Cloudie [adapted from 15]

Service discovery agent (SDA), a cloud ontology, a database of cloud services, multiple cloud crawlers and a web interface together constitutes 'Cloudle'. Whenever cloud service providers register their services in the Cloud database, cloud crawlers assist them in building and maintaining databases in the 'Cloudle'. A web interface is used by consumers to enter their queries. Queries can be consumer's functional, technical and budgetary requirements for cloud services.

SDA consists of four sub-modules which are named as follows:

- a query processor,
- a service reasoning module,
- a price and time slot matching module and
- a service rating module.

These modules help Cloudle in the service discovery process [15]. The query processor helps SDA excerpt essential keywords from consumers' queries. SDA then performs service reasoning with the help of cloud ontology. Service reasoning can be similarity reasoning, compatibility reasoning and numerical reasoning. Also, SDA finds the level of matching between the price and schedule constraints of consumers and providers with the help of the time and slot module. These help Cloudle satisfy its requirements.

Similarity Reasoning [15] helps in determining an increased chance of finding an appropriate service by matching services requested by consumer to other similar services. For example, when buying a product of particular size from a seller, if the required size match is not found, this service helps in finding products within a similar size range or another product from any other seller. Similarity Reasoning determines the degree of similarity between two concepts 'x' and 'y' by finding their commonly accessible shared concepts.

Let, the two set of nodes $\alpha(x)$ and $\alpha(y)$ share some nodes reachable by both the nodes 'x' and 'y'. Then, their degree of similarity can be calculated using the equation below:

$$\text{sim(x,y)} = \left| \alpha(x) \cap \alpha(y) \middle/ \alpha(x) \right|$$

The above equation represents a measurement of commonalities between two nodes 'x' and 'y' based on the perspective of 'x'. Similarly, the degree of similarity can also be calculated in the context of 'y' as shown in equation below:

$$\text{sim(y,x)} = \left| \alpha(x) \cap \alpha(y) \middle/ \alpha(y) \right|$$

In Compatibility Reasoning [15], compatibility between two sibling nodes in cloud ontology is obtained. For Example: Comparing two nodes of different versions of operating systems like Windows7 and Windows8 and evaluating the compatibility between them. Since, SDA cannot compare two versions of similar software, this reasoning measures the compatibility by finding the degree of difference based on the chronological orderings of different versions of software and determining the degree of similarity between them.

Numerical Reasoning [15] helps SDA to determine similarity between two numeric concepts like memory size and processor speed relative to its maximum and minimum size.

Price and Time Slot Matching [15] assists SDA in searching for cloud providers that have available time slots in their cloud services which matches as per consumers' requirements. This checks for the least possible price difference and between the acceptable range of prices of the consumer and provider.

Service Negotiation Phase

In order to achieve service-level agreements, messages have to be exchanged. This is achieved in the service negotiation phase where message exchange occurs between consumers and brokers, and between brokers and providers [15]. Market-oriented approach and Cloud negotiation model are used in this phase.

Market-oriented approach [15] is based on market model which consists of resource/service provider agents, consumer agents, and broker agents. Broker agents are responsible for service negotiation phase which accept service requests made by number of consumers via consumer agents. Broker agents also buy resources from provider agents. Those agents compose collection of resources from many different providers to form a bundled service that can be subleased to other consumers as requested. Each agent can negotiate with another based on the service contracts through service level agreements (SLA). SLA is an agreement that contains all the details of the service agreed to between them and penalties upon violation of those agreements.

In Cloud negotiation model [15] as shown in Figure 7, many-to-many negotiation approach is observed between consumer agents and broker agents. Each broker agent can accept service requests from many consumer agents. Also, each consumer agent can submit its requests and requirements to multiple broker agents.

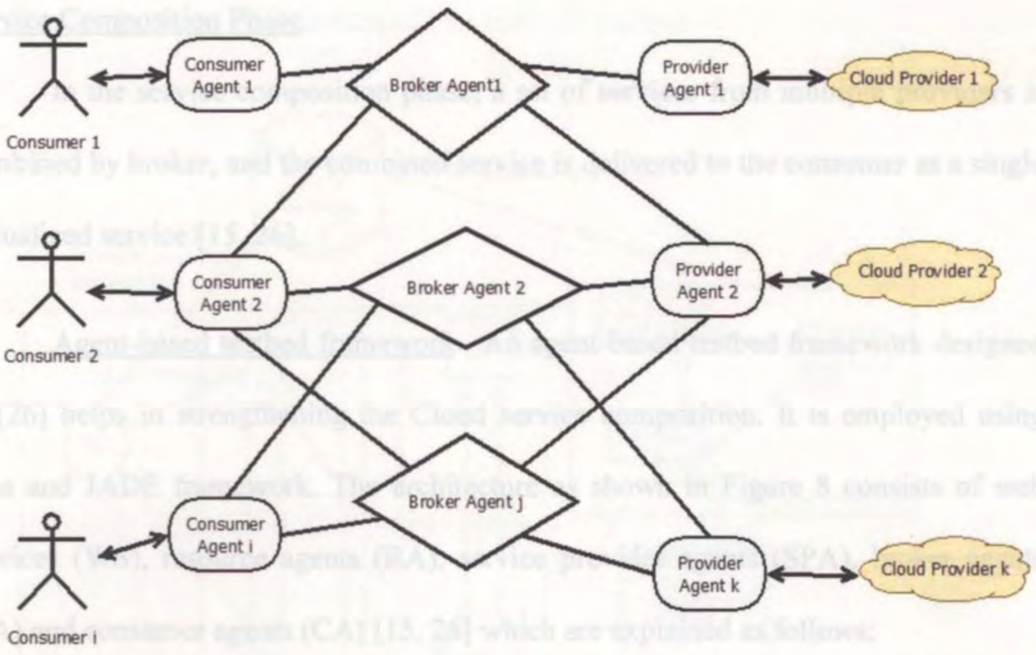

Figure 7: Agent Based Testbed [adapted from 15]

However, negotiation between broker agents and provider agents is observed to be adopting concurrent one-to-many mechanism as shown in Figure 7. Each cloud resource market is composed of multiple cloud providers that can provide many different types of cloud resources, i.e., Quality of Service such as varying in price, response time, trustworthiness and data quality. It is seen that each broker agent can negotiate in many such resource markets simultaneously in order to reach an agreement with one of those providers.

Service Composition Phase

In the service composition phase, a set of services from multiple providers is combined by broker, and the combined service is delivered to the consumer as a single virtualized service [15, 26].

Agent-based testbed framework. An agent-based testbed framework designed in [26] helps in strengthening the Cloud service composition. It is employed using Java and JADE framework. The architecture as shown in Figure 8 consists of web services (WS), resource agents (RA), service provider agents (SPA), broker agents (BA) and consumer agents (CA) [15, 26] which are explained as follows:

Web Service (WS): It provides an interface to software applications or to resources like databases present in the system. This interface helps Resource Agents (RA) to access the required service. The software applications can also be remotely accessed by Service Provider Agents (SPA) via RAs. Each web service performs different functions based on the requirement of the service. The parameters defined for the required service acts as an input while the respective result obtained after performing the required service acts as an output to the web service [26].

For example (example similar to [26]), if requirement of consumer is Storage space needed then capacity of CPU, type of CPU is input to WS. In this case, output is initial empty storage space, space left after service fulfillment, and CPU identifier.

The service location is determined by URL of address.

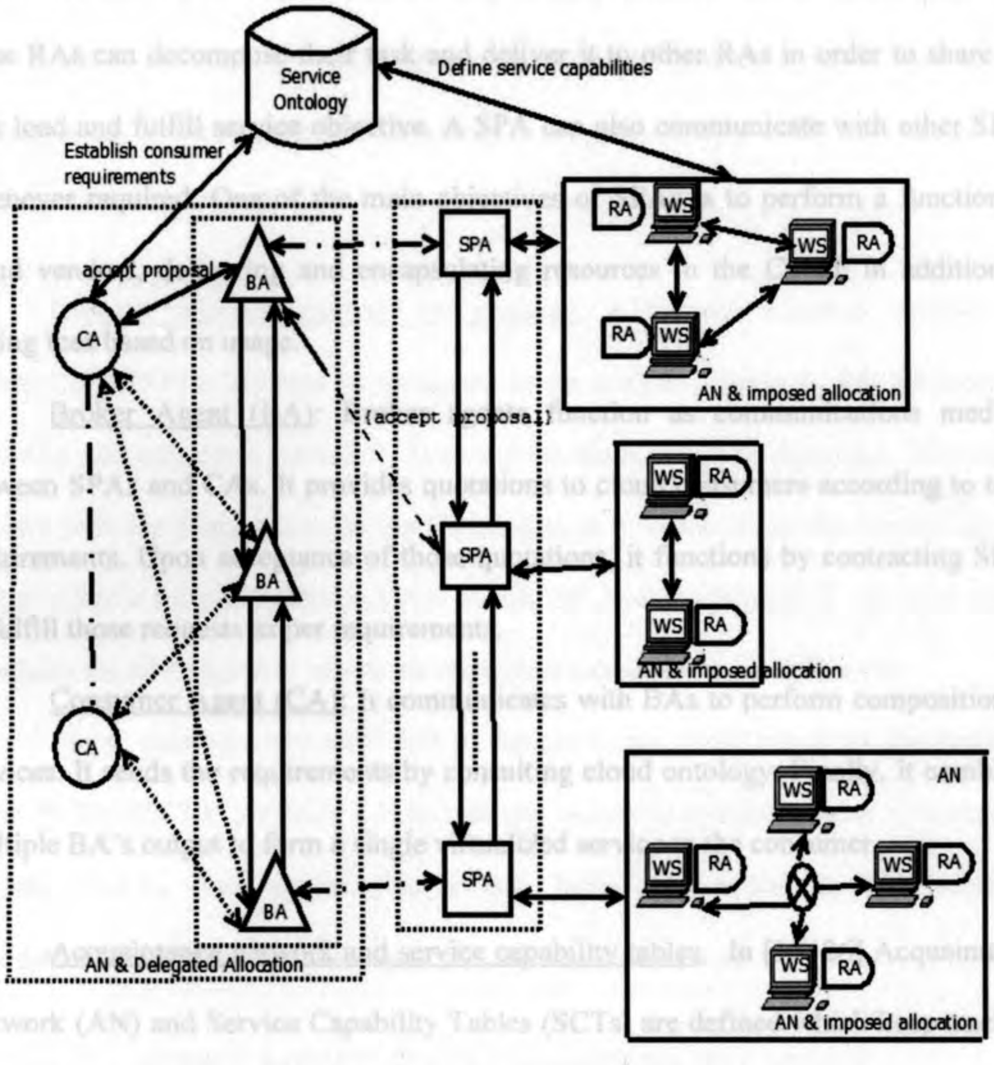

Figure 8: Multi-agent System Architecture Simulating Cloud Service Composition [adapted from 26]

Resource Agent (RA): Managing and controlling an access to WS is done by RAs. It receives requests from other RAs or service providers and conveys necessary service messages back to the requester.

Service Provider Agent (SPA): It assists in managing and controlling a set of RAs. RAs that are managed by same SPA usually perform similar tasks. However, these RAs can decompose their task and deliver it to other RAs in order to share the task load and fulfill service objective. A SPA can also communicate with other SPAs whenever required. One of the main objectives of SPAs is to perform a function of cloud vendors, delivering and encapsulating resources in the Cloud, in addition to setting fees based on usage.

Broker Agent (BA): Broker agents function as communications medium between SPAs and CAs. It provides quotations to cloud consumers according to their requirements. Upon acceptance of those quotations, it functions by contracting SPAs to fulfill those requests as per requirements.

Consumer Agent (CA): It communicates with BAs to perform composition of services. It sends the requirements by consulting cloud ontology. Finally, it combines multiple BA's output to form a single virtualized service to the consumer.

Acquaintance network and service capability tables. In [15, 26] Acquaintance Network (AN) and Service Capability Tables (SCTs) are defined which keep track of agents in the Cloud. Here, agents can be Consumer Agent, Broker Agent or Service Provider Agent. Each agent has their own *Acquaintance Network*. AN is a matrix. Its columns records list of other agents (that an agent knows). Its rows record their service capabilities. For instance, each CA can maintain a SCT of several BAs. AN is responsible for keeping records of agents (CA, BA, SPA) service capabilities in the Cloud while SCT keeps records of state of agents as well as information about their service capabilities. Various states that any agent in the Cloud can be in are as follows: *'available', 'unreachable', 'failed'* and *'busy'*. Whenever an agent interacts with any other agent in an agent testbed as shown in Figure 8, its state changes from one to another as per service processed. This leads to frequent changes in the information that is stored in SCT of cloud.

Focused selection contract net protocol. A focused selection contract net protocol (FSCNP) is adopted by all agents in the testbed (Figure 8). FSCNP helps in selecting relevant cloud services [15] as per the requirement of consumer. It interacts directly with the other agents in the Cloud system in order to get the service needed and so called a focused selection CNP. In FSCNP, by consulting SCT, an agent easily identifies the other agent to which the requested message is to be delivered.

Since, messages are only sent to the particular agent based on the result of using SCTs, FSCNP provides a very efficient means to reduce number of messages exchange among cloud agents. This, in turn, helps cloud system to focus more on relevant cloud services in order to provide the required service.

MOBILE AGENTS FOR WEB SERVICES DISCOVERY AND SELECTION

One of the most important applications on the Internet is Web services discovery and selection. The growing use of web services has led to a thought that the traditional mechanism of service discovery must be reconsidered. So, a new Cloud Computing architecture combined with a new Mobile Agent system for discovery of web services has been introduced in [19]. It creates a mechanism to choose and to access the web facilities that deal with the similarities between the requests of the clients and the description of web services offered. The proposed framework is referred to as "Open Cloud Computing Federation Based on Mobile Agents" (OCCF and MABOCCF) [19]. A new matching algorithm for web services selection is also discussed in this section to obtain the discovered web services with more precision.

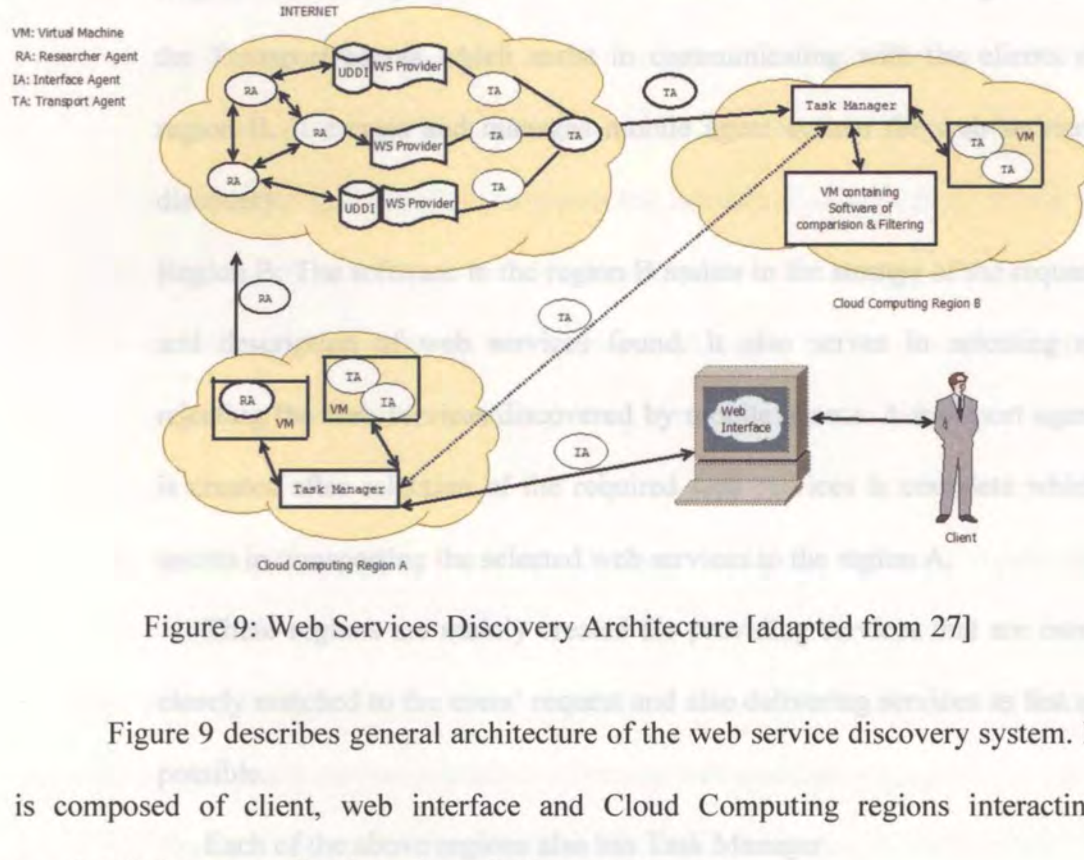

Figure 9: Web Services Discovery Architecture [adapted from 27]

Figure 9 describes general architecture of the web service discovery system. It is composed of client, web interface and Cloud Computing regions interacting together in internet to facilitate the web discovery process.

The architecture of such discovery system consists of three parts as explained in [19, 27]:

1. <u>Web Interface Part</u>: This part contains web interface which allows clients to access the region A of cloud. List of actions includes the following: creating a login account, providing an interface for displaying the results, etc.
2. <u>Cloud Computing Part</u>: It consists of two regions based on the models of Software-as-a-Service (SaaS) and Storage as a Service (SAaS).

   Region A: This region deals with the creation of the Interface agents and the Transport agents which assist in communicating with the clients of region B. It creates and manages mobile agent system for web services discovery.

   Region B: The software in the region B assists in the storage of the request and description of web services found. It also serves in selecting or rejecting the web services discovered by mobile agents. A transport agent is created after selection of the required web services is complete which assists in transporting the selected web services to the region A.

   These regions are mainly created for providing services that are more closely matched to the users' request and also delivering services as fast as possible.

   Each of the above regions also has Task Manager.

   <u>Task Manager of Region A</u>

   Task Manager (TM) of each of the regions functions as receiving the client's requests, associates it with the identifier in the form of a code so that each client's request can be identified uniquely. Based on the key words extracted from the request, TM of region A creates researcher mobile agents. These agents, then, do syntactic research of web services. Knowledge Base (KB) present in the task manager contains synonyms of each word which helps in updating the knowledge of researcher agents and facilitates in web services discovery.

<u>Task Manager of Region B</u>

In case of TM of region B, the request and its respective web services description obtained from researcher agent is transmitted to the software of comparison and filtering along with the lemmas of each of those words. A software of comparison and filtration functions in selecting the best web services that corresponds to the request. In region B, TM creates transport agent which functions in transporting each selected web service to region A.

3. <u>Mobile Agents Part</u>: It consists of Interface Agent, Transport Agent and Research Agent.

Interface Agent transmits the requested service of client to region A. It also displays the service selected to client via web interface.

Transport Agent transports client requests from region A to B as well as the discovered web services back to region A.

Researcher Agent serves in performing a syntactical research on the basis of keywords.

The scenario of steps as shown in Figure 9 is described as follows [27]:

When client makes a request via web interface, creation of an interface agent (IA) takes place at Region A. Task Manager at Region A then facilitates the delivery process by creating a set of researcher mobile agents. Mobile Agents carry with it keywords of the request made along with a unique identifier. A transport agent (TA) is finally created which transports the service to the Region B of cloud.

Researcher mobile agents are launched to further assist in providing the requested service. Researcher mobile agent makes the keywords based search. Upon finding a service that matches the request, it creates a transport agent which delivers the service along with the identifier to Region B of cloud via Task Manager.

In Region B, Task Manager introduces a software of comparison and filtration which functions in selecting the best web services that corresponds to the request. This software also rejects the unwanted services which is later deleted from Region B by task manager.

On the basis of resources used in the process, cost of filtering operation is calculated by Task Manager in Region B. The selected service and cost are transported to Region A with the help of Transport agent. In Region A, global cost is calculated by Task Manager along with the services that have high degree of acceptance. Interface Agent (IA) is then created for displaying of the selected web services as well as the billing information for the client via Web Interface.

This approach of web service discovery in Cloud Computing benefits from a "Business Model". Business model facilitates an instantaneous access to cloud services, at the same time, also ensures an availability of the resources requested by the client.

This section presents a new Cloud Computing architecture and matching algorithm for web services discovery and selection that is benefitted by the characteristics of mobile agents.

MOBILE AGENT FOR CLOUD SERVICE MIGRATION

Service Migration in Hybrid Cloud

When migration of services and cloud applications occur from a busy cloud to an idle cloud, it is necessary to ensure an efficient way for better performance in the hybrid cloud environment. Hybrid cloud has ability to manage resources both internally and externally in an efficient and effective manner. Though hybrid cloud uses various cloud service providers like Amazon and Google to enter the private network, it still faces several challenges while interacting in the public environment. Some of them are: determining the type of jobs that can be migrated, determining the perfect environment, timings and the way it is migrated to the public cloud. Hence, maintaining a good service migration policy is an important aspect to be considered in a hybrid cloud environment.

A prototype integrating private cloud with public cloud is proposed in [4] which helps in achieving automatic, intelligent migration between hybrid clouds. This prototype exploits Mobile Agents to manage all resources, monitor system behavior, and negotiate all actions of the hybrid cloud [4]. It comprises of federated broker and the hybrid cloud. Federated broker assists in solving problems based on semantics heterogeneity (i.e., varying ways in which data is organized in various components). Hence, this prototype can create new jobs dynamically by initiating computing clusters from public cloud while loading of private cloud is in progress.

The architecture also has a Federated broker layer and several mobile agents interacting in the system as shown in Figure 10.

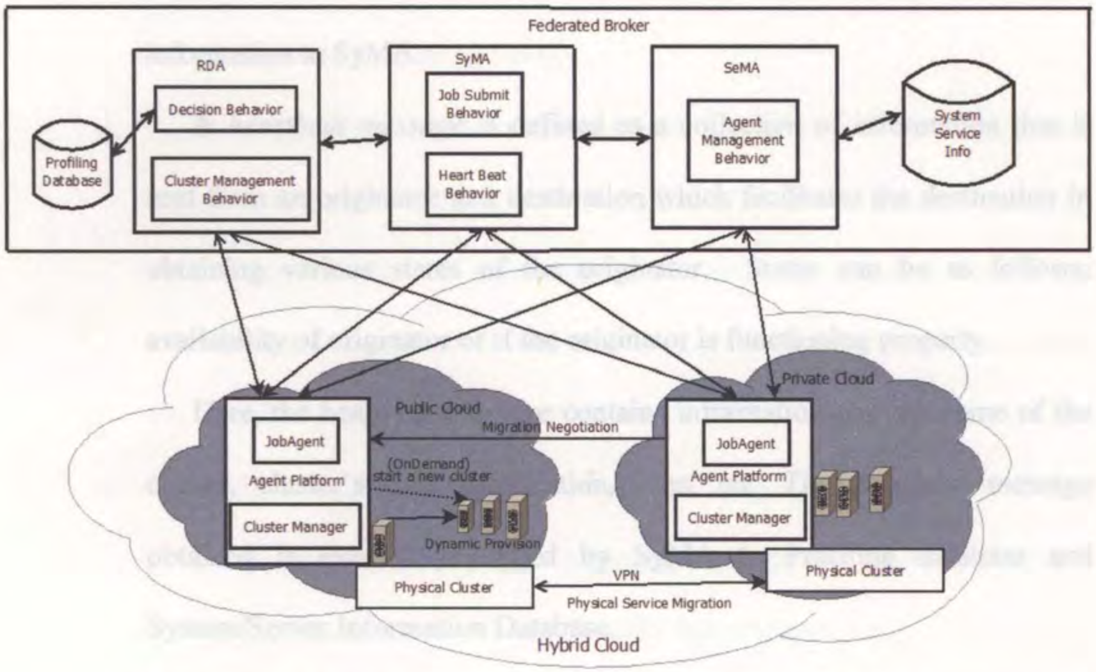

Figure 10: General Architecture of the System [adapted from 4]

Federated Broker Layer [4]: This layer is responsible for observing and controlling behaviors of resources, reconfiguring system scale as well as migration of services and balancing loads between clouds. This layer is composed of several components, which are as follows:

System Monitoring Agent (SyMA), Reconfiguration Decision Agent (RDA), Service Migration Agent (SeMA), Profiling Database and System/Service Repository.

1. System Monitoring Agent (*SyMA*): The agent is responsible for monitoring activities in the hybrid cloud. It computes the loading based on the resource related information of nodes present in the hybrid cloud. This agent can either communicate actively with the *Cluster Admin Agent* (*CAA*) present in the cluster or can passively obtain a heartbeat message that has loading information to SyMA.

   A *heartbeat message* is defined as a collection of information that is sent from an originator to a destination which facilitates the destination in obtaining various states of the originator. States can be as follows: availability of originator or if the originator is functioning properly.

   Here, the heartbeat message contains information like the name of the cluster, cluster's load information, jobs, etc. The heartbeat message obtained is eventually logged by SyMA to Profiling database and System/Server Information Database.

2. Reconfiguration Decision Agent (*RDA*): The RDA functions by reconfiguring and adjusting the Cloud environment based on the information obtained from SyMA. It retrieves the most recent status and loading data in the cluster and scales the public cloud accordingly. For example, if a private cloud is seen overloading by SyMA, it contacts RDA which functions by initiating the reconfiguration process. In turn, RDA

maintains the balance of loading between private and public cloud. Public cloud is also scaled down by RDA if the loading of the private cloud is observed to be under loaded during a time period.

3. <u>Service Migration Agent (*SeMA*)</u>: This agent deals with migration of service between private and public cloud. According to the policy of service migration decided between clouds, SeMA initiates the service migration process after it receives the message from RDA informing it that the public cloud is to be scaled up.
4. <u>Profiling Database</u>: This component helps in storing all the information related to the public cloud, including number of virtual machines, size of memory and storage, the bandwidth of link to public cloud, and the usage of these resources. This information can be utilized when deciding to scale the Cloud.
5. System/Service Repository: This component keeps the information related to the system and service. The information can be the location of private and public clouds, services running in the hybrid cloud. etc.

<u>Agents in the Cloud</u>: Besides the above components, Cluster Admin Agent (CAA) and Job Agent (JA) are also present in both private and public cloud platform.

1. <u>Cluster Admin Agent (*CAA*)</u>: It is present in each cluster (not shown in Figure 10) and assists in managing the whole cluster. It is responsible for all negotiation activities that occur with the agents in the federated layer.

2. <u>Job Agent (*JA*)</u>: This agent is responsible for delivery of services to other clouds. Any service that needs to be migrated to another cloud is encapsulated into a job agent. It contains information such as service's execution file, service's input data, location to be migrated, etc.

Hence, this section mainly focuses on how these agents work together for service migration flow. The steps can be summarized as follows [4]:

- A heartbeat message containing the loading information is periodically sent to SyMA from CAA.
- Computing of the loading information is done by SyMA in order to obtain the loading status of the Cloud. If it is unbalanced, SyMA informs RDA which acts on the public cloud to adjust its size.
- The CAA of public cloud is sent the decision information from RDA which initiates new clusters for receiving and deploying migrated service, and finally sends the result back to RDA.
- RDA then informs SeMA to migrate the service to public cloud.
- According to the service migration policy, SeMA determines which service should be migrated to the public cloud. It then informs both private and public cloud.

<u>Service Migration Policy</u>: The policy of service migration is used to decide when to initiate the service migration process. Some of the algorithms used in this prototype are based on three concepts [4] namely: Job Count (JC), Size of Job (SJ) and Estimated Finish Time (EFT).

Job Count (JC): In this case, a decision is made based on the number of jobs in the private and public clouds.

$$JC_{PR} - JC_{PU} \geq T_C$$

where if JC_{PR} and JC_{PU} represent number of jobs in the private and public cloud respectively, then T_C is a threshold value. According to this policy, SeMA will pick the $(JC_{PU} + T_C + 1)^{th}$ job from the private cloud which is triggered to be migrated to the public cloud.

Size of Job (SJ): Here, total size of jobs in both clouds becomes the deciding factor for initiating the migration action. If the sum of size of jobs in the private cloud is larger by a threshold than that in the public cloud, then according to the policy of SeMA, it will pick a job in the private cloud to be migrated to the public cloud.

Estimated Finish Time (EFT): In this case, the migration decision is based on the estimated finish time of jobs in the private and public clouds. If the sum of finish time of all jobs in private cloud is larger by a threshold than that of the public cloud, then according to service migration policy, SeMA selects a job in the private cloud to be migrated to the public cloud.

Service as an Agent Service (SaaAS)

This section introduces another architecture known as 'Service as an Agent Service' (SaaAS) that helps in migration of service in cloud [18]. SaaAS is a Mobile Agent based service developed for a Cloud Computing system to work in an internet environment, i.e., Wide Area Network (WAN). Since every cloud has limited storage space, it can move its data to other neighboring clouds when demands on the storage are not sufficient to handle more information. SaaAS helps to store user's information in the Cloud without notifying the users about the location where their data is stored and how it can be accessed. Similarly, it can very effectively retrieve the data from the neighboring clouds and return it to the user. SaaAS makes use of Mobile Agents to facilitate the software and data migration service and helps in reducing the heavy communication overhead in the Internet.

SaaAS architecture is a combination of a number of domains. Each domain is a cloud that is composed of several users interconnected using high bandwidth LAN. Each domain is linked to another domain through a low bandwidth WAN. This architecture facilitates in sharing overheads of migration and hence avoids a single central server system which may cause a bottleneck in the system.

Two important concepts in a Cloud Computing system are Software as a Service (SaaS) and Data as a Service (DaaS) [18]. These are combined together to become a Mobile Agent in SaaAS. These mobile agents facilitate the data sharing process and hence enables SaaAS to become more flexible, adaptable and usable. It also makes it more suitable to work in the Internet than a conventional Cloud Computing system, where a user has to install new software in order to retrieve their information.

1. Software as a Service (SaaS): It hosts various software and applications as a service which is also accessible to customers across the internet. Hence, it avoids the necessity to install and run the software in their local computer.

2. Data as a Service (DaaS): Mobile agents make data and information available to its customers and users in the network as a service. These data are present in various formats and from multiple sources which can be easily accessed as if users are obtaining it from their own local machine or accessing it from the internet semantically, i.e., retrieving data based on meaning of keywords.

Besides migration of software and data in the internet, mobile agents in SaaAS also functions as a system management.

The cooperation between the agents in several domains can efficiently maintain the data coherence between domains and can also improve the service migration process. As shown in Figure 11, all agents in SaaAS are classified as: Interface agent, Working Agent, Domain Manage Agent and Main Management Agent. Functions of each agent are explained below with the help of Figure 11.

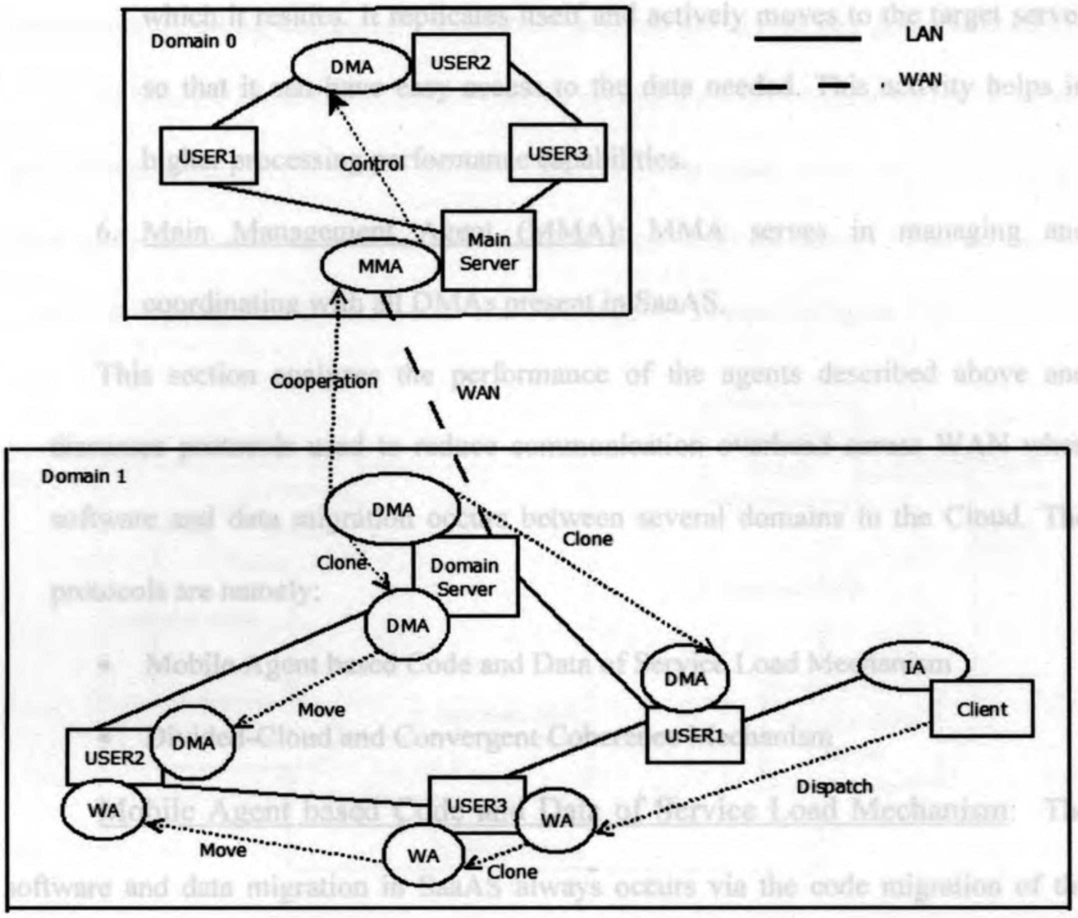

Figure 11: Work Sharing and Collabortion of Agents in SaaAS
[adapted from 18]

3. <u>Interface Agent (IA)</u>: IA resides on the user host and interacts with the user. IA then coordinates with other agents to perform the computing task.
4. <u>Working Agent (WA)</u>: WA after accepting instructions from IA, combines the software and data together to send it to the target domain for execution. It also returns the result of execution to the client.

5. Domain Manage Agent (DMA): DMA serves as managing the domain in which it resides. It replicates itself and actively moves to the target server so that it can have easy access to the data needed. This activity helps in higher processing performance capabilities.
6. Main Management Agent (MMA): MMA serves in managing and coordinating with all DMAs present in SaaAS.

This section analyzes the performance of the agents described above and discusses protocols used to reduce communication overhead across WAN when software and data migration occurs between several domains in the Cloud. The protocols are namely:

- Mobile Agent based Code and Data of Service Load Mechanism
- Divided-Cloud and Convergent Coherence Mechanism

Mobile Agent based Code and Data of Service Load Mechanism: The software and data migration in SaaAS always occurs via the code migration of the agent. This section analyzes performance of agents in different domains of the Cloud. Runtime codes are the library codes of language used that are executed when a program is running and is required to implement various features of the language. According to the observation presented in Figure 12, load latency decreases if all runtime code is involved in running agents in different domains of the Cloud. However, a high network communication overhead is observed at this time. But when an agent loads runtime code directly from code server, network communication overhead is found to be decreasing and load latency becomes high as explained below in Figure 12. Based on the observation of relationship between load latency and the runtime code loading mechanism, a protocol is discussed in [18] and presented below.

In SaaAS, Main Code Server (MCS) contains all the runtime codes of agent. So, Domain Code Server (DCS) contacts MCS, gets the codes and stores them in their local buffer for their use. If an agent running in client's device wants to load the codes, according to the protocol, it can follow the six steps as shown in Figure 12.

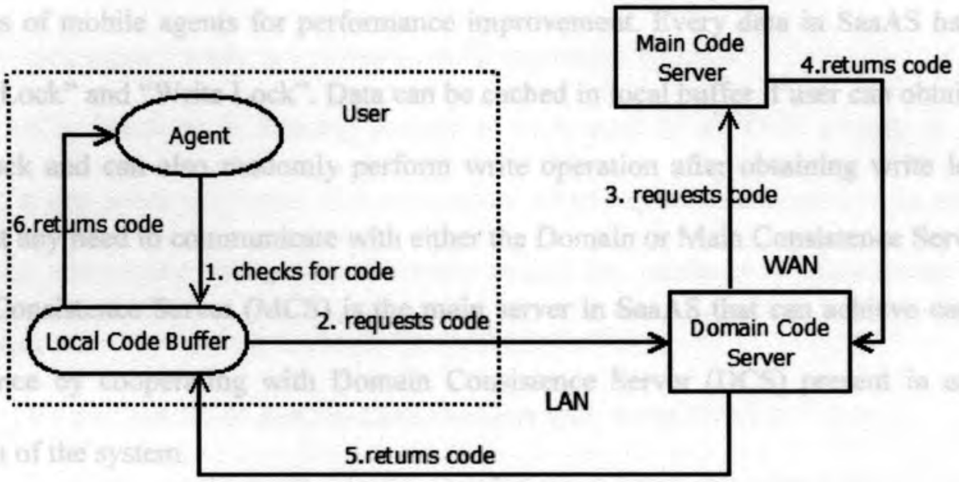

Figure 12: Loading Mechanism of Runtime Codes [adapted from 18]

If the codes are already cached in local buffer on user's machine then only step (1) and (6) are required. If codes are loaded from local domain server across LAN then it has to undergo steps (1), (2), (5) and (6). Latency is much lower in this case than loading codes over WAN when codes need to be accessed from Main Server following all six steps as shown in the Figure 12. This protocol discussed above follows a *dynamic hierarchical code loading mechanism*. This loading mechanism can help mobile agents perform more effectively and at its best since it reduces heavy communication overhead across the internet.

Divided-cloud and convergent coherence mechanism. SaaAS follows Divided-Cloud and Convergent Coherence Mechanism (DCCM) in order to access data in the Cloud Computing system [18]. Since SaaAS uses UNIX semantics, this mechanism helps SaaAS to reduce communication overhead in WAN as well as utilize the unique features of mobile agents for performance improvement. Every data in SaaAS has a "Read Lock" and "Write Lock". Data can be cached in local buffer if user can obtain a read lock and can also randomly perform write operation after obtaining write lock without any need to communicate with either the Domain or Main Consistence Server. Main Consistence Server (MCS) is the main server in SaaAS that can achieve cache coherence by cooperating with Domain Consistence Server (DCS) present in each domain of the system.

DCS functions in bringing all the lock requests from all users in its own domain. This activity of DCS can help in minimizing the communication across WAN and also improves the performance of cache coherence management. DCS contains a lock table which keeps track of all the read and write locks that it maintains from all the users in its domain. If any user 'A' in a domain requests a 'read lock' for data 'X', then DCS checks its 'lock table' to see if it contains the lock needed. If it finds that another user in the same domain has the lock, it assigns it to user A. It then updates it's 'lock table'. For this, DCS simply copies the cache of data from buffer of another user to the user A without any need to contact MCS for the lock. However, it contacts MCS when the required lock is not present in the table.

If many users in a single domain make a read or write request for data X at the same time, then the DCS converges all similar requests from different users in its domain and sends only one request to MCS. Upon receiving the read or write lock from MCS, any other read or write lock request and buffer copying activity from other users in the same domain is then processed in its own domain. Hence, it is observed that any lock request made in a domain can be converged by DCS.

MCS functions in keeping records of lock state of all DCS present in the system. It also sends read/write lock to domains when requested and updates its table. This lock converging ability can efficiently reduce the overhead of maintaining the lock state.

This mechanism of applying and releasing lock is explained in Figure 13.

If a UserC wants to put a 'write lock' on data X and the users User E, UserF and UserG contain the 'read lock' for that data, then the actions that take place are summarized below [18]:

1. An agent 'A' is created by UserC, which is dispatched to DCS1 in order to apply the write lock on X.
2. Upon agent 'A' arrival at DCS1, DCS1 checks the record from its lock table to see if any user has the write lock on X. After discovering no matching lock is contained in the table, 'A' moves to MCS where it applies the write lock to MCS.

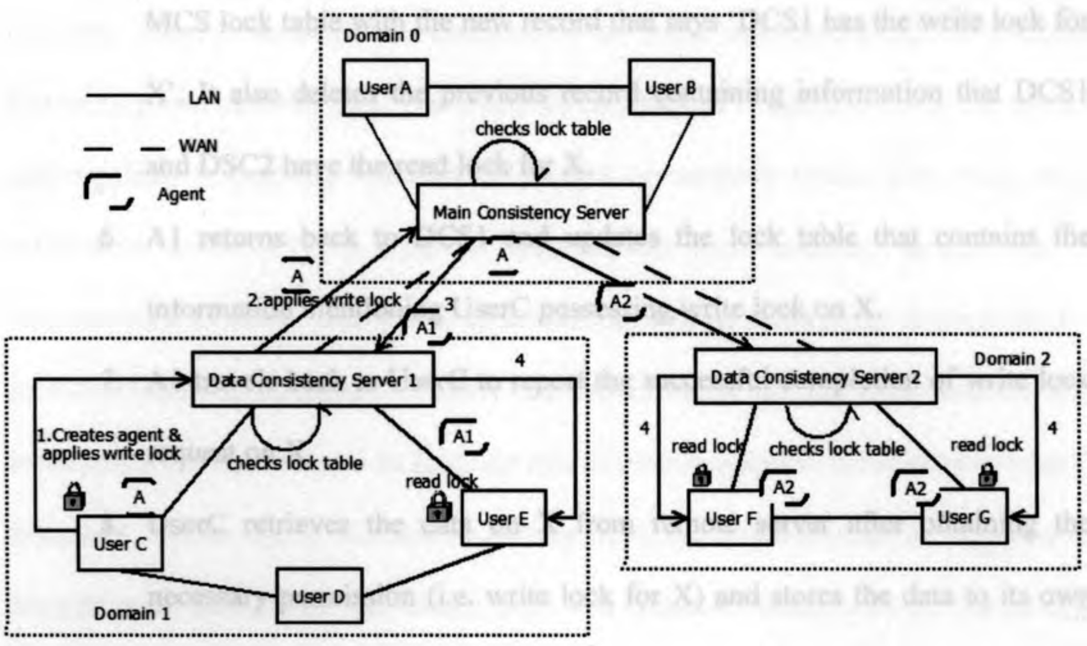

Figure 13: Domains in Divided Cloud for Representing Convergent Coherence Mechanism [adapted from 18]

3. 'A' checks the lock table on MCS. It finds two different domains DCS1 and DCS2 containing the read lock of D. So, agent replicates itself into A1 and A2 which move to domains DCS1 and DCS2 respectively.

4. At DCS1, it finds UserE from the lock table containing read lock for X. Hence, A1 moves to UserE and requests it to release the read lock. UserE releases the read lock and sends acknowledgement to A1 which returns back to DCS1 and then to MCS.

   Same process is repeated for A2 where User F and User G release read lock and send acknowledgement to A2 which returns back to DCS2 and eventually A2 moves to MCS.

5. At MCS, A2 terminates after transferring its information to A1. A1 updates MCS lock table with the new record that says 'DCS1 has the write lock for X'. It also deletes the previous record containing information that DCS1 and DSC2 have the read lock for X.
6. A1 returns back to DCS1 and updates the lock table that contains the information mentioning UserC possessing write lock on X.
7. A1 travels back to UserC to report the successful completion of write lock request on X.
8. UserC retrieves the data on X from remote server after obtaining the necessary permission (i.e. write lock for X) and stores the data to its own local buffer. Since, X can be easily accessed from local buffer. It eliminates the need for read/write operation on data X from server. Hence, it is observed that mobile agents' ability to replicate, move and cooperate in various environment of cloud helps in minimizing communication overload in WAN and the cache coherence management overhead in DCS and MCS.

MOBILE AGENT FOR CLOUD SCALABILITY

Cloud Computing is a concept that deals with sharing data and computations over a scalable network of computers, spanning across end user computers and data centers that provides web services. Data center is a collection of virtualized computers, in a network, that are distributed geographically [28]. In Cloud Computing, delay in service might occur due to longer distances between data centers. High service load in virtual machines may also result in increased response time to the customer. Hence, scalability is observed as the most challenging task in a Cloud Computing environment [28]. The unknown behavior of consumers using the service and regularly fluctuating volumes of service invocations makes this issue more difficult to tackle [29]. In order to make service more scalable, service cost of response time of server and quality of service have to be maintained. A good service provider becomes necessary to serve customers based on the user location and workload of the data centers. In [28], the role of mobile agents in cloud environment is discussed. Mobile agents function in moving from one network to another in order to provide more scalable service to end users and without limiting network bandwidth. Scalability makes cloud service more flexible where users can easily request the service needed without knowing about the methodology applied for delivering their service. For example, if a storage capacity of any cloud becomes full then it may share storage with other neighboring clouds. Upon users' request for any data that is unavailable, it may easily obtain its data from other clouds having the desired resources.

There are several approaches for maintaining the scalability of Cloud Computing. However, this section mainly focuses on agent-based framework for achieving high scalability [28]. The process involves two steps. Searching for the nearest cloud that can satisfy the request when cloud becomes overloaded is the first step. While searching for the least response time of a virtual machine (VM) for the closest datacenter is the second.

Depending on the usage and demand of application, cloud scalability has the ability to dynamically increase or decrease the number of server instances. Hence, this section describes an algorithm for implementing the agents (Cloud Mobile Agent and Directory Agent) involved to facilitate the process and make Cloud Computing more scalable.

The architecture presented in [28] consists of a hybrid cloud with a number of mobile agents containing information about resources in the Cloud. Whenever any service is requested in the Cloud, mobile agents look for available resources and make decisions based on the free and allocated resources. Various mobile agents present in this architecture that help in obtaining scalability are as follows:

- Cloud Mobile Agent (MA_C): It is present in every cloud and functions as maintaining resource information and keeps a record of their availability at any particular time.
- Directory Agent (DA_C): It keeps track of the capacity of clouds as well as information about all the Cloud mobile agents (MA_C) present in the Cloud. Cloud Mobile Agents of every cloud created should be registered within Directory Agent which helps in providing scalability in service by maintaining a database of all the registered Cloud Mobile Agents.

Whenever a new cloud is created, a MAc registration request is sent to the closest DAc. The request contains the Cloud information with which it is related (private/public/hybrid cloud). DAc then sends an acknowledgement signal indicating that the request has been received and MAc is registered in its system.

When a service is requested in a cloud that does not contain any more available resources, these mobile agents act in providing scalability to the Cloud. The communication between agents in the Cloud for scalability is explained below:

- MAc in the Cloud requesting for service becomes activated and contacts the nearest DAc to obtain the list of other MAc that can offer the requested service.
- Upon receiving the request, the directory agent looks into its database and returns the list of available cloud mobile agents MAc s.
- Service request is sent to those available MAc s by initiator MAc. Initiator MAc then waits for their reply.
- The Cloud containing the required resources responds back to the initiator MAc as soon as it is ready for delivering the services.
- All the received responses are scanned by the initiator. It negotiates with all those MAcs that responded in order to get a good deal in terms of the low cost and faster processing capabilities. After deciding on the most suitable MAc, it assigns the service request to it.

Hence, this process of interaction between mobile agents in the Clouds results in providing a scalable and efficient Cloud Computing system.

Sometimes, devices with limited capacity become so overloaded that it is not able to proceed even in the presence of Mobile Agents. Hence, this section presents a platform where various Mobile Agents can execute even with limited cloud capacity. In [30], a platform is developed that has the capability to support various mobile agents based on the capacity of the system where those mobile agents reside. Several mobile clients with different configuration are present in the platform in order to handle activities performed by mobile agents. Platform functions in checking the configuration of mobile clients for its ability to handle mobile agents. Mobile Agents might either contain both code and data to the server or only the information about state of the client. Based on the result obtained regarding a client's ability to handle mobile agents, mobile agents are sent to the server either with the state alone or with both the code and data together. State alone is sent in the case when the server already contains the necessary code. Transferring mobile agents to remote machine for data collection and computation can help in minimizing battery power consumption as well as avoiding the slow and error-prone connections in the network.

In [30] mobile clients are differentiated into thick and thin clients depending on their capacity. In order to provide scalability, proper Mobile Agents chosen based on the information it contains, traverse across the network in different ways for each of its mobile clients. This process helps mobile clients with minimal capacity and resources to achieve higher processing results.

The architecture developed consists of Thick and Thin Clients, Decision Maker, Extractor, System Repository, Agent joiner/ splitter, Agent initiator, Mobile Agent Dispatcher as shown in Figure 14. These are described below [30]:

a. Thick Mobile Clients: Clients with enough resources are known as thick mobile clients. These clients have the capacity to handle both the code (in a database) and data together in mobile agents and send it to the server for execution. This helps processing of code done in a remote powerful machine away from the mobile device at which thick mobile client runs. Finally, the result of execution is returned to the mobile client. Hence, it reduces the execution load in the mobile device which in turn saves the power of mobile devices in use. Mobile agents used in these clients are known as *Thick Mobile Agent.*

b. <u>Thin Mobile Clients</u>: Configuration of Thin mobile clients does not have the capacity to handle execution of particular application. In these clients, Mobile agents can only send data to the server. This data represents the 'state' of the client. A simple Graphical User Interface (GUI) is present to manipulate this data. In this case, since only data is sent to the server via mobile agent, server already stores code for the execution. Hence, execution of application occurs even in the absence of support for the resources in the client. This also helps avoiding of excessive use of resources for code dispatch to the server since mobile agents performs the task of code dispatch efficiently. Mobile agents used in these clients are known as *Thin Mobile Agent.*

c. <u>Decision Maker:</u> Based on the efficiency of the client, some threshold value for decision making is specified. The values obtained from capacity of memory, speed, java support, etc., that are already contained in the database of the system help in determining if data transmission is to be performed.

d. Extractor: This functions in extracting the client capabilities to deliver either both the code and data together or only the state of the client. This information is available in the initiator as shown in Figure 14.

e. System Repository: It consists of information of agents and associated clients. Client information can be information of client device capabilities which can help in selecting proper client. This information can assist the mobile agents to traverse to the direction of the proper client.

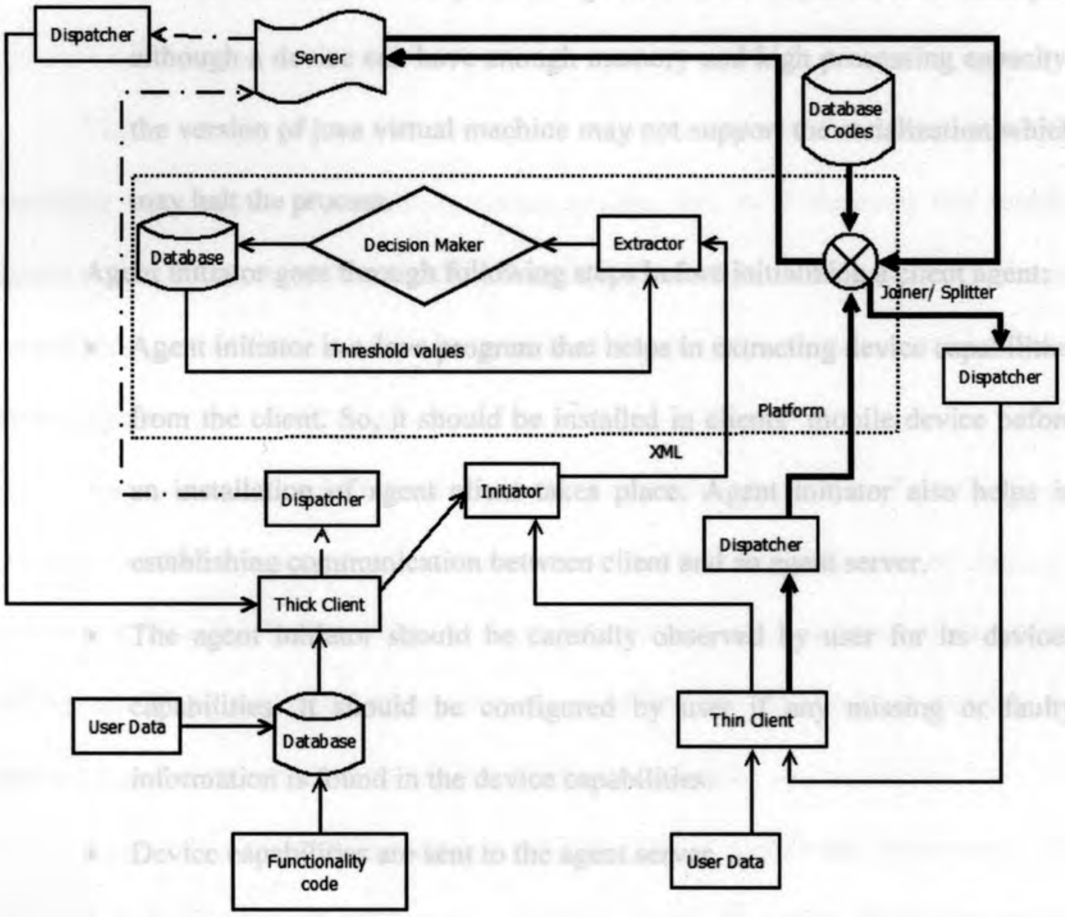

Figure 14: Architecture for Achieving Scalability [adapted from 30]

f. Mobile Agent Dispatcher: It dispatches Mobile Agents from Client to Server for sending code and data, and also from server to client to pass the result back from the server to client's machine.

g. Agent Initiator: It initiates the mobile agent clients for the first time. In the process, it also introduces the capabilities of client devices of their properties that include CPU speed, execution memory, storage, Java Virtual edition and the version used. It is very important to mention the edition and version of java along with other features. For example, although a device can have enough memory and high processing capacity, the version of java virtual machine may not support the serialization which may halt the process.

Agent initiator goes through following steps before initializing a client agent:

- Agent initiator is a Java program that helps in extracting device capabilities from the client. So, it should be installed in clients' mobile device before an installation of agent client takes place. Agent initiator also helps in establishing communication between client and an agent server.
- The agent initiator should be carefully observed by user for its devices capabilities. It should be configured by user if any missing or faulty information is found in the device capabilities.
- Device capabilities are sent to the agent server.

- A suitable agent client is delivered based on the capabilities of the device. It is sent directly using the agent initiator or indirectly using a communication medium such as email of HTTP-downloads.

h. Agent joiner/splitter: This part of the architecture splits data and code of the mobile agents for the clients that are incapable of running mobile agents locally (due to client capabilities). Agent splitter is the only way to differentiate between thick and thin clients. It makes information ready for the dispatch and combines data and code together upon receiving it from the client.

Figure 14 illustrates the process of migration of mobile agents with the assistance of initiator, dispatcher, extractor, splitter, etc. It is observed that mobile agents with their capacity to work and adapt in different kinds of clients (and clients' capabilities) can efficiently assist in execution of code in a remote machine. Thus, Mobile agents assist in achieving scalability.

Some other related works that can help in scalability issues of mobile agent platform in Cloud Computing environment are as follows [30]: use of software frameworks like JADE (JAva Development Framework), Mobile Information Agents (MIA), etc. These can also help mobile agents to perform efficiently at run-time by moving agents from one node to another whenever required.

JAva DEvelopment Framework (JADE) [30] is a software framework that provides a middleware to multi-agent systems. It can be easily distributed across various machines which can help avoid the need to share the same operating systems.

It has graphical tools that allow agents to interact with different types of client devices with its limited features to support any application. JADE platform can help mobile agents to change configuration at runtime and whenever required. Thus, it helps mobile agents to interact even in the limited cloud capacity. Hence, JADE assists in improving the scalability issue in cloud.

Also, use of Mobile Information Agents (MIA) [30] can help in achieving scalability in the Cloud. It is an intelligent information system that collects information via GPS-receiver or mobile phone tracker. The information collected can help web browser in client's mobile device to share its information to the clients.

MOBILE AGENT BASED SECURE CLOUD COMPUTING

Security is an important factor in any communication environment. Since users are unaware of the location of their data storage in the case of Cloud Computing, it becomes essential that service providers should maintain trust in data transmission and make cloud services more reliable to users. In the current cloud architecture, it is easy for attackers to invade the virtualized environment of cloud. Hence, to overcome the problem of data integrity and to provide monitoring of users' data and also to maintain a more secure environment, mobile agent based architectures are developed. These are discussed in the following sections.

Ensuring Trust in Cloud

Trust is defined as a degree of belief between two parties or people. It is a very important factor in Cloud Computing. Cloud Service Provider (CSP) and the Cloud

Customer should always maintain a trustworthy relationship between them [5]. Customers requesting service may sometimes interfere in other people's request. Determining a magnitude of trust in order to identify a good customer is a major challenge in the field of Cloud Computing. Keeping this in mind, a mobile agent based approach of ensuring trustworthiness in the Cloud is presented in this section. The architecture as shown in Figure 15 below consists of Cloud Broker, Cloud Customer and Cloud Service Provider [5].

Cloud Customer: Customers refer to those persons who visit the cloud and make a request for service needed in the Cloud.

Cloud Service Provider: It refers to organizations that offer services to cloud customers from a remote location (i.e., datacenter) through the Internet. All service providers must be registered with the Cloud broker to ensure trustworthiness.

Cloud Broker: It is a trusted third party in the Cloud that is responsible for the monitoring and control. It determines a trustworthy relationship between the customers and the Cloud service provider by continuously monitoring their activity and in turn helps in eliminating any malicious entity found in the Cloud. Trust is ensured based on penalties or prizes given to CSP based on their behavior as monitored and reported by mobile agents to Cloud Broker. A sub-entity of a cloud broker known as 'Key Provider' functions by assigning both private and public keys to the customer in the Cloud. The Cloud broker is also responsible for maintaining an agent pool which assists in adding and dropping agents dynamically based on customers' requests [5].

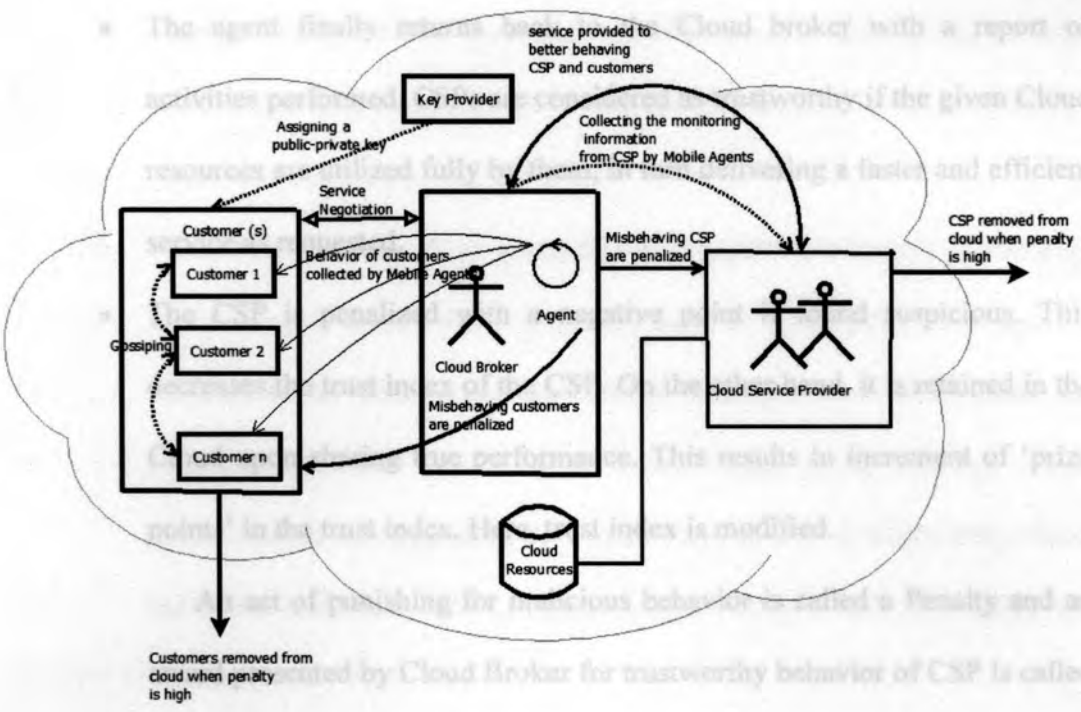
service provided to better behaving CSP and customers
Key Provider
Assigning a public-private key
Collecting the monitoring information from CSP by Mobile Agents
Service Negotiation
Customer (s)
Customer 1
Customer 2
Customer n
Gossiping
Behavior of customers collected by Mobile Agents
Agent
Cloud Broker
Misbehaving customers are penalized
Misbehaving CSP are penalized
Cloud Service Provider
CSP removed from cloud when penalty is high
Cloud Resources
Customers removed from cloud when penalty is high

Figure 15: Agent Based Trust Mechanism [adapted from 5]

The sequence of steps for ensuring trust as shown in Figure 15 is explained below:

- A pair of keys (private and public) is allotted to each customer by 'Key Provider' (a sub-entity of Cloud Broker) as soon as it enters the Cloud.
- To be qualified as a service provider, each organization is required to get registered with the Cloud broker.
- To guarantee the trustworthiness of cloud service provider, a dummy job request is initially sent by the broker to the CSP. Then, Cloud broker invokes one of its agents (AgentA) from agent pool to monitor those instances of virtual machine that are instantiated by the CSP.

- The agent finally returns back to the Cloud broker with a report of activities performed. CSPs are considered as trustworthy if the given Cloud resources are utilized fully by them, in turn delivering a faster and efficient service as requested.
- The CSP is penalized with a negative point if found suspicious. This decreases the trust index of the CSP. On the other hand, it is retained in the Cloud upon sharing true performance. This results in increment of 'prize points' in the trust index. Here, trust index is modified.

  An act of punishing for malicious behavior is called a Penalty and an award presented by Cloud Broker for trustworthy behavior of CSP is called a Prize.

  The penalty is applied to CSP whenever CSP is found suspicious. The process of penalizing continues until the tolerance threshold of the Cloud broker is reached. Each suspicious activity reduces the threshold and finally the CSP is eliminated from the Cloud believing the entity to be a malicious one.
- When the execution of customer's job is finished, customers share their experiences among themselves via 'Gossiping'. A Gossip contains an USER_EXPERIENCE field which indicates the experience of the customer who is responsible for message origination. Its value is either set to +1 (satisfied customer) or -1 (dissatisfied customer).

- Gossip messages when combined together assist in calculating the trust index of the CSP since it helps in determining each customer's experience after their jobs are processed in cloud.
- Cloud Broker assists in the process of ensuring trustworthiness of customer by invoking a second agent (AgentB) from an agent pool. This agent is responsible for monitoring the Gossip messages from customers. The malicious customer usually sends a misleading gossip message which might hamper the reputation of a CSP. Even if CSP is trustworthy, these malicious customers send gossip messages setting USER_EXPERIENCE field as -1.

  In this case, Cloud broker could easily determine the customer to be a malicious one since it is already familiar with the status of the CSP and would know that it is trustworthy. (This is explained above where Cloud Broker obtained the report from returning agent, AgentA, after invoking its AgentA). True experience can be determined when Cloud broker knows the CSP is trustworthy, by invoking AgentA, and also a gossip message from a customer is set with its User Experience field as +1.

  Hence, if a customer is determined to be malicious, it is penalized with a penalty point that reduces the trust index of the customer. The customer is finally removed from the Cloud after it crosses the tolerance threshold. Prize points are awarded to customers sharing true experience that results in increasing the value of the trust index of that customer.

Before any customer message is sent to the Cloud Broker, the message is always encrypted with its own private key. When Cloud Broker receives the message, it is then forwarded to its specified destination. Hence, trust is ensured through the use of messages which are regularly monitored by the agents in the cloud.

The mechanism as shown in Figure 15 also helps to avoid Message-Replay attack and Man-in-the-middle attack.

In *Message-Replay attack* [31], a valid data flow is maliciously repeated or delayed either by originator itself or an attacker who tries to infect the transmission of data and IP packets in a network.

Man-in-the-middle attack [32] is a kind of eavesdropping where the victim gets an illusion of having a direct private connection with the other party. So, when message transfer occurs between them, the attacker invades into their communication by controlling the entire conversation activity and hence makes an independent connection with the victim.

Security Agents

In [16], a trust model for cloud architecture is proposed which assures security at multiple levels in the Cloud infrastructure. Mobile Agents in this model act as Security Agents which help in obtaining necessary information from virtual machine which can be useful to Cloud Service Provider and users to maintain privacy of their data and virtual machines. Security agents can dynamically migrate and autonomously replicate as per requirements and can even audit data for security purposes and do event logging. (Hence, Mobile Agents ensure integrity and authenticity of virtual machines.)

In the architecture presented in [16] security is implemented at three checkpoints in the Cloud environment. These are located at the entry point of the communication level in the Cloud network, at Xen hypervisor in virtual machine and at each individual instance of virtual machine assigned to each user. Xen hypervisor is a component of Xen virtual environment that has been utilized for study of security agents in [26]. Besides this, Xen virtual environment also consists of Domain 0 (DOM0), Domain U (DOMU) and Domain Management and Control (Xen DM&C) unit.

Xen Hypervisor: It is an interacting medium between hardware and operating systems. It is also responsible for management of CPU scheduling and memory partitioning activities of several virtual machines.

Domain 0 (DOM0): It is a virtual machine in which Linux kernel is modified so as to interact with and handle requests from other virtual machines (DOMU PV and HVM Guests) running on the hypervisor.

Domain U (DOMU): DOMU consists of para-virtualized (PV) Guest and fully virtualized (HV) Guest virtual machines.

To accomplish a trust model, the architecture as shown in Figure 16 consists of three mobile agents namely Mobile Agent1 (MB1), Mobile Agent2 (MB2) and Mobile Agent3 (MB3) [16].

Mobile Agent1: This agent is present at the entrance of cloud and is directly controlled by client. It facilitates in establishing a secure communication and data exchange between client and the service provider and also manages cloud resources from the client side.

Mobile Agent2: It resides in DOM0 virtual machine. It monitors activities of CSP and infrastructure provider in DOM0. Upon observing any abnormal behavior, it reports it to MB1, which informs the client for further action. This agent is also responsible for managing the integrity of drivers and applications executing in DOM0. Mobile Agent3: This agent is deployed at each instance of the client's virtual machine and keeps track of each event that occurs in DOMU virtual machine. It also monitors behaviors of other shared virtual machines and DOM0 for any malicious activities that might occur in them. A notification is sent to MB1 if any anomalous activity is detected, hence providing the client a control over data.

Communication between agents is entirely secure and encrypted. The details of activities as that occur in this architecture are shown in Figure 16. They are described as follows [16]:

Step 1: After an authentication of client and cloud service provider is performed and SSL key is generated for secure communication between agents, Mobile Agent1 from client side is sent to the service provider site.

Step 2: Activation of Mobile Agent1 occurs which establishes a new private session key with the client side so as to maintain a secure communication. Based on the requirement and service load, Mobile Agent1 requests data and resources from the cloud service provider for client side. MB1 also monitors resources as well as performs regular checks on CSP for data misuse.

Step 3: Based on the request made, CSP allocates virtual machines and other resources for client. At this point, a Mobile Agent2 is created and transferred to the platform by CSP where it receives new resources. This agent monitors authenticity and reliability of software, drivers and DOM0 platform where it resides. In addition to this, Mobile Agent2 regulates behavior of DOM0, performs registration of all drivers and hardware for verification and authentication.

Step 4: Mobile Agent3 is found in each virtual machine associated with the client (DOMU). At first, Mobile Agent3 registers itself to Mobile Agent1 and creates a new pair of keys with Mobile Agent1. It helps in establishing a secure communication between virtual machines and Mobile Agent1 on the client side and also performs regular monitoring of behavior of all synchronized applications.

Step 5: In this step, when connection is closed by user, all the allocated resources are withdrawn and the link is disconnected.

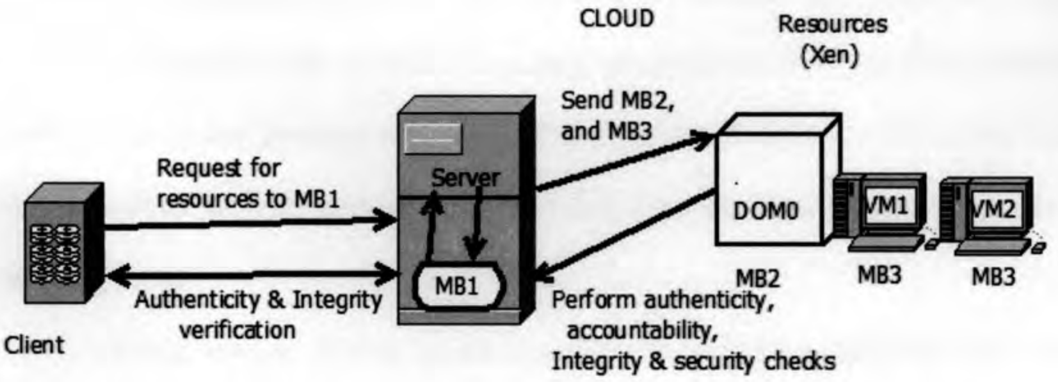
CLOUD
Resources (Xen)
Send MB2, and MB3
Request for resources to MB1
Server
DOM0
VM1
VM2
MB1
MB2
MB3
MB3
Authenticity & Integrity verification
Perform authenticity, accountability, Integrity & security checks
Client

Figure 16: Work Flow Model [adapted from 16]

Hence, privacy and security of client data is maintained using this model. In this model, client gains control over his own data using security agents present in the system so that it becomes easier for them to maintain privacy and security of their information. Cloud service provider can regulate security policies thereby avoiding virtual machine attacks. It is also observed that mobile agents play an important role in building a trust based communication environment between various entities.

Chapter 6

CONCLUSION

Since Mobile agents have characteristics of intelligence and mobility, it can be used to implement Cloud Computing. This paper studied the advantages of Mobile Agents in Cloud Computing environment and presented several areas where mobile agents are in action with Cloud Computing. Mobile Agents have a wide scope in the field of Cloud Computing. It is possible to overcome various shortcomings of Cloud Computing by integrating mobile agents in computing paradigm.

The contributions of this paper are mentioned as follows:

A different perspective of Cloud Computing technology delivering fast, efficient and more reliable service in the presence of mobile agents is presented. Firstly, Cloud Computing is introduced along with its various architecture; service models and deployment models. The introductory section also mentions several obstacles present in the Cloud Computing environment like providing effective performance in the presence of number of applications in the same computing node, making service scalable to users in addition to making the service available to users in any APIs used.

In next section, Mobile Agent is introduced so as to explain reasons behind combining it in the Cloud Computing environment. A brief comparison of Cloud

Computing with Grid Computing and Cluster computing is presented. Grid and Cloud Computing are explained as more convincing models than Cluster in terms of standardizing APIs, security, interoperability, new business models, and dynamic pricing systems for complex services. An integration of cloud on grid architecture based on Mobile Agent, called as Cloud Agency, is introduced. Cloud Agency can dynamically add and configure services on virtual clusters utilizing various features of mobile agents. The high computation and secure platform infrastructure of Grid and the standard model and technologies of Cloud are combined providing final users with full administrative control over their information stored.

Finally, a major area is surveyed in this paper which includes various fields in Cloud Computing environment where interactions of Mobile Agents have facilitated the process. Several architectures are presented which shows an effective use of Mobile Agents. 'Cloudle' architecture is used for discovering service requested by customer. Various approaches like market oriented approach and cloud negotiation model used for achieving service level agreements is covered in the service negotiation phase. In service composition phase, a set of services from multiple providers is combined, and the combined service is delivered to the consumer as a single virtualized service by the use of mobile agents. A framework is used which provides a very efficient means to reduce number of messages exchange among cloud agents. To ensure an efficient way for service migration in hybrid cloud environment, mobile agents are used in a prototype which has helped in achieving automatic, intelligent migration of services. This prototype has provided an efficient service by using mobile agents to create new jobs dynamically by initiating computing clusters from public cloud while loading of private cloud is in progress. Various policies are defined based on which migration of services occurs in cloud.

Similarly, SaaAS architecture is explained which makes use of Mobile Agents to perform various service load mechanisms in facilitating software and data migration services which has aided in reducing heavy communication overhead in internet. Use of mobile agents in achieving scalability in cloud is also explained in this paper. Mobile agents have assisted in reducing service cost of response time of server in providing scalable service. An algorithm is explained in increasing or decreasing the number of server instances dynamically with the help of mobile agents as well as an architecture developed is presented in order to show importance of mobile agents in making cloud service more scalable. Besides these advantages of mobile agents in Cloud Computing technology and a trust based service model is also discussed in this paper in order to maintain trust in data transmission and make cloud services more reliable to users.

In conclusion, Mobile Agents have been an essential communication medium for Cloud Computing technology due to its wide range of features. Being an autonomous, auto-adaptable, mobile and intelligent entity, it has played a vital role in Cloud Computing environment. Its features when utilized can bring a revolutionary change in the world of internet and can introduce a new era in the field of Cloud Computing.

REFERENCES

[1] M. Armbrust et al., "Above the clouds: A Berkeley view of cloud computing," UC Berk. Rel. Adap. Dist. Sys. Lab., Berkeley, CA, Tech. Rep. UCB/EECS-2009-28, Feb. 2009. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html [Accessed: 06 April 2014].

[2] "Cloud Computing", Mar. 22, 2013. [Online]. Available: http://www.wikinvest.com/concept/CloudComputing [Accessed: 3 Apr., 2014].

[3] N. Mirzaei, "Cloud computing," Comm. Gr. Lab., Indiana University, Indianapolis, IN, Rep. 390, 2008. [Online]. Available: http://cgl.soic.indiana.edu/publications/ReportNarimanMirzaeiJan09.pdf [Accessed: 06 April 2014].

[4] C. Fan et al., "Agent-based service migration framework in hybrid cloud," *IEEE International Conference on High Performance Computing and Communications,* September [illegible] Banff, AB. pp. 887-892. Available: IEEE Xplore 2011, http://www.ieee.org [Accessed: 10 September 2013].

REFERENCES

[5] A. Ramaswamy et al., "A mobile agent based approach of ensuring trustworthiness in the cloud," *IEEE-International Conference on Recent Trends in Information Technology*, Chennai, Tamil Nadu, pp. 678- 682, June 3-5, 2011.

[6] J. Woo, "Introduction to cloud computing," *the 10th KOCSEA 2009 Symposium* UNLV, (2009) December 18-19. Available: http://www.dialogic.com/~/media/products/docs/whitepapers/12023-cloud-computing-wp.pdf [Accessed: 17 Sept. 2013].

[7] B. L. Sahu and R. Tiwari, "A comprehensive study on cloud computing," *International Journal of Advanced Research in Computer Science and Software Engineering,* vol. 2, no. 9, Sept. 2012.

[8] A. Singh and M. Malhotra, "Analysis for exploring scope of mobile agents in cloud computing," *International Journal of Advancements in Technology*, vol. 3, no. 3, pp. 172-183, July 2012. [Online]. Available: http://draartisingh.com/paper/Analysis%20for%20exploring%20scope%20of%20mobile%20agents%20in%20cloud%20computing.pdf [Accessed: 25 Aug. 2013].

REFERENCES

[1] M. Armbrust et al., “Above the clouds: A Berkeley view of cloud computing,” UC Berk. Rel. Adap. Dist. Sys. Lab., Berkeley, CA, Tech. Rep. UCB/EECS-2009-28, Feb. 2009. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html [Accessed: 06 April 2014].

[2] “Cloud Computing”, Mar. 22, 2013. [Online]. Available: http://www.wikinvest.com/concept/CloudComputing [Accessed: 3 Apr., 2014].

[3] N. Mirzaei, “Cloud computing,” Comm. Gr. Lab., Indiana University, Indianapolis, IN, Rep. 390, 2008. [Online]. Available: http://cgl.soic.indiana.edu/publications/ReportNarimanMirzaeiJan09.pdf [Accessed: 06 April 2014].

[4] C. Fan et al., “Agent-based service migration framework in hybrid cloud,” *IEEE International Conference on High Performance Computing and Communications,* September 2-4, 2011, Banff, AB. pp. 887-892. Available: IEEE Xplore 2011, http://www.ieee.org [Accessed: 10 September 2013].

[5] A. Ramaswamy et al., “A mobile agent based approach of ensuring trustworthiness in the cloud,” *IEEE-International Conference on Recent Trends in Information Technology*, Chennai, Tamil Nadu, pp. 678- 682, June 3-5, 2011.

[6] J. Woo, “Introduction to cloud computing,” *the 10th KOCSEA 2009 Symposium,* UNLV, (2009) December 18-19. Available: http://www.dialogic.com/~/media/products/docs/whitepapers/12023-cloud-computing-wp.pdf [Accessed: 17 Sept. 2013].

[7] B. L. Sahu and R. Tiwari, “A comprehensive study on cloud computing,” *International Journal of Advanced Research in Computer Science and Software Engineering,* vol. 2, no. 9, Sept. 2012.

[8] A. Singh and M. Malhotra, “Analysis for exploring scope of mobile agents in cloud computing,” *International Journal of Advancements in Technology*, vol. 3, no. 3, pp. 172-183, July 2012. [Online]. Available: http://draartisingh.com/paper/Analysis%20for%20exploring%20scope%20of%20mobile%20agents%20in%20cloud%20computing.pdf [Accessed: 25 Aug. 2013].

[9] H. Saxena et al., “Cloud computing,” *International Journal of Emerging Technology and Advanced Engineering*, vol. 2, no. 4, pp. 348-351, April 2012. [Online]. Available: www.ijetae.com/files/Volume2Issue4/IJETAE_0412_59.pdf [Accessed: 06 April 2014].

[10] H. I. Syed et al., “Survey on cloud computing,” *International Journal of Emerging Technology and Advanced Engineering*, vol. 3, no. 4, pp. 308-311, April 2013.

[11] G. Lewis, “Basics about cloud computing,” Carnegie Mellon University, Pittsburgh, PA,White paper, Sept. 2010. [Online]. Available: [http://resources. sei.cmu.edu/asset_files/WhitePaper/2010_019_001_28877.pdf]_[Accessed: 06 April 2014].

[12] P. Mell and T. Grance, "The NIST definition of cloud computing", *National Institute of Science and Technology Special Publication*, pp. 800-145. Sept. 2011. [Online]. Available: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf [Accessed: 30 Oct., 2014].

[13] N. Sadashiv and D. Kumar, “Cluster, grid and cloud computing: A detailed comparison,” in *Proceedings of the 6th International Conference on Computer Science & Education,* ICCSE 2011, Aug. 3-5, 2011. SuperStar Virgo, Singapore. pp. 477-482.

[14] W3C, “PROV-AQ: Provenance access and query,” June 2012. [Online]. Available: http://www.w3.org/TR/2012/WD-prov-aq-20120619/ [Accessed: 30 Oct. 2014].

[15] K. M. Sim, “Agent based cloud computing,” *IEEE Transactions on Services Computing,* vol. 5, no. 4, pp. 564-577, Oct-Dec 2012.

[16] P. Hada et al., “Security agents: A mobile agent based trust model for cloud computing,” *International Journal of Computer Applications,* vol. 36, no. 12, pp. 12-15, Dec.2011.

[17] A. Ali et al., “Implementation of cloud computing approach based on mobile agents” *International Journal of Computer and Information Technology,* vol. 2, no. 6, pp. 1141-1149, Nov. 2013.

[18] G. Chen et al., “SaaAS - The mobile agent based service for cloud computing in internet environment,” in *Proceedings of the 2010 Sixth International Conference on Natural Computation (ICNC 2010),* vol. 6, pp. 2935-2939, 10-12

Aug. 2010, Yantai, Shandong. Available: IEEE Xplore, http://www.ieee.org [Accessed: 20 Jan. 2010].

[19] S. Hamza et al., "A new cloud computing framework based on mobile agents for web services discovery and selection," in *Proceedings of the 13th Arab Conference on Information Technology*, pp. 587-594, Dec.10-13 2012. Available: http://www.acit2k.org/ACIT/2012Proceedings/12524.pdf [Accessed: 28 Aug. 2013].

[20] "Comparison of grid computing vs. cluster computing," *Journal of Theoretical and Applied Information Technology*. [Online]. Available: http://www.jatit.org/research/introduction_grid_computing.htm [Accessed: 24 Aug. 2013].

[21] "Difference between cloud, cluster and grid computing," *Cloud Computing Competence Center for Security*. [Online]. Available: http://www.aisec.fraunhofer.de/content/dam/aisec/Dokumente/Publikationen/Studien_TechReports/englisch/studieCloudComputingSicherheit-AISEC-en.pdf [Accessed: 24 Aug. 2013].

[22] I. Foster et al., "Cloud computing and grid computing 360- degree compared," in *Proceedings of the Grid Computing Environments Workshop, GCE 2008,* pp. 1-10, Nov. 2008, Austin, TX. [Online]. Available: IEEE Xplore, http://www.ieee.org [Accessed: 17 Sept. 2013].

[23] G. Lewis, "The role of standards in cloud-computing interoperability," in *Proceedings of 46th Hawaii International Conference on System Sciences (HICSS),* 2013, pp. 1652-1661.

[24] R. Aversa et al., "Cloud agency: A mobile agent based cloud system," in *Proceedings of the 2010 International Conference on Complex, Intelligent and Software Intensive Systems, CISIS 2010, 15-18 Feb. 2010, Krakow, Poland.* Available: IEEE Xplore, http://www.ieee.org. [Accessed: 10 Sept. 2013].

[25] J. Kang and K. M. Sim, "Cloudle: An agent-based cloud search engine that consults a cloud ontology," in *Proceedings of International Conference on Cloud Computing and Virtualization,* CCV 2010, Singapore, pp. 312-318, May 2010.

[26] J. Octavia, Gutierrez-Garcia, and K. Sim, "Self-organizing agents for service composition in cloud computing," in *Proceedings of the 2010 IEEE Second International Conference on Cloud Computing Technology and Science,* pp. 59-66, Nov-Dec, 2010.

[27] S. Hamza et al., "A cloud computing approach based on mobile agents for web services discovery," in *Proceedings of the 2012 Second International Conference on Innovative Computing Technology,* INTECH 2012, 18-20 Sept. 2012, Casablanca, Morrocco. pp. 297-304. Available: IEEE Xplore, http://www.ieee.org. [Accessed: 9 Oct. 2013].

[28] A. Singh and M. Malhotra, "Agent based framework for scalability in cloud computing," *International Journal of Computer Science & Engineering Technology,* vol. 3, no. 4, pp. 41-45, Apr 2012.

[29] J. Y. Lee and S. D. Kim, "Software approaches to assuring high scalability in cloud computing," in *Proceedings of 7th International Conference on E-Business Engineering,* ICEBE 2010. 10- 12 Nov. 2010, Shanghai, China. pp. 300-306.

[30] Dr. K. P. Kaliyamurthie and A. Antonidoss, "Intelligent mobile agents for heterogeneous devices in cloud computing," *International Journal of Scientific & Engineering Research,* vol. 4, no. 4, pp. 1353-1358, Apr. 2013.

[31] "Message-Replay attack", Nov. 7, 2014. [Online]. Available: http://en.wikipedia.org/wiki/Replay_attack.

[32] "Man-in-the-middle_attack," Nov. 7, 2014. [Online]. Available: http://en.wikipedia.org/wiki/Man-in-the-middle_attack.