The Future for Adaptive Software Development in Cloud Computing Environment Using Multi Agent System

Dr. Abeer Tariq AL_Obaidy
Computer Sciences Department, University of Technology/ Baghdad.
Email:abeer282003@yahoo.com
Mayada M.Shihab Al_Doori
Computer Sciences Department, University of Technology/ Baghdad.
Email:mayadamuhammed2014@yahoo.com

Received on: 23/4/2014 & Accepted on: 4/12/2014

ABSTRACT

Cloud computing system provides large-scale infrastructure for high performance computing. The convergence of interests between multi-agent systems that need reliable infrastructures and cloud computing systems that need intelligent software with dynamic, flexible and autonomous behavior can result in new systems and applications.

This paper presents a proposed system using the intelligent multi-agent system in cloud computing .When it comes to developing the way of thought in a given environment, it is essential to think on a critical level in order to reach a state of understanding and operating on a level that best suits the needs of the said environment through evaluating the status of a situation and working to reach a decision or take an action.

The work presented aims to reach this state of smart decision-making that is essential to software developers when they are laying the foundations of the software internalization. The proposed system pioneers led the use of a multi-agent system in cloud computing, as there was no such initiative before in the development of software. The system proposed introduces a means for the autonomous decision-taking and critical thinking that is necessary to develop and evaluate the social behavior, universality and adaptive roles that the XML rational agent which is responsible for the cloud reservoir is capable of.

Keyword : Cloud Computing, Ontology, Multi-agent Systems, Proposed System.

المستقبل التكيفي لتطوير البرمجيات في بيئة الحوسبة السحابية عن طريق نظام متعدد الوكلاء

الخلاصة

يقدم هذا البحث مقترحا لنظام يتكل على نظام العملاء المتعددين (المبرمجين) في الحوسبه السحاب عندما يتعلق الامر بتطوير طريقة تفكير الفرد في بيئة ما, لا بد من التفكير بمستوى حساس و نقدي من اجل الوصول

إلى حالة من الفهم و العمل على مستوى يلائم حاجة البيئة المذكورة أنفا من خلال تقييم الحالة و العمل للوصول إلى حل أو لاتخاذ إجراءا ما .

هذا العمل يهدف لكي يصل إلى هذه الحالة من إتخاذ القرار الذكي و التي تعد أساسية بالنسبة لمطوري البرمجيات عندما يقومون بوضع الأسس للتدخيل البرمجي . و يعد هذا العمل الأول من نوعه بأستعمال نظام متعدد العملاء (المبرمجين) في الحوسبه السحابيه, إذ لا يوجد عمل يسبقه في مجال تطوير البرمجيات. يقدم هذا النظام المقترح وسيلة لاتخاذ القرار الذاتي و التفكير النقدي بصورة ذاتيه و التي تعد اساسية (XML) لتطوير و تقييم السلوك الاجتماعي و الصفة العالمية و الأدوار التكيفية التي يدعمها العميل المنطقي مستودعات السحابه.

INTRODUCTION

Cloud computing systems provide large-scale infrastructures for high performance computing that can adapt to user and application needs. They can be integrated with Multi-agent systems (MASs) which is capable of intelligent behavior. Then make agents take a proper decision automatically. So that it can get high-performance and making clouds more flexible and autonomic.

Cloud Computing

Cloud computing provide elastic services, high performance and scalable data storage to a large and everyday increasing number of users. The National Institute of Standards and Technology (NIST) have given a complete reference definition. NIST defined " Cloud computing is a pay-per-use model for enabling available, convenient, on- demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, services) that can be rapidly provisioned and released with minimal management effort or service provider interaction."

Moreover, “Cloud model promotes comprised of five key characteristics, three delivery models, and four deployment models.”

Delivery Models

Infrastructure as a service (IaaS):

offering virtualized resources (computation, storage, and communication) on demand is known as Infrastructure as a service(IaaS). So that the consumer may deploy and run arbitrary software, including operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems; storage, deployed applications, and possibly limited control of select networking components (e.g., host firewalls).

Examples of IaaS service providers are: Amazon elastic Compute Cloud (EC2), Clever, Eucalyptus, GoGrid, FlexiScale, Linode, Nimbus, Open Nebula, PerfCloud,RackSpace Cloud, and Terremark. As shown in figure (1).

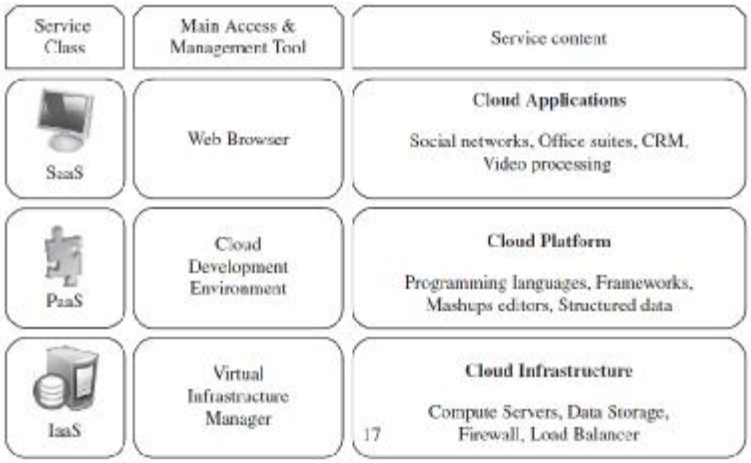


Figure (1) Cloud Computing Services

Platform as a service (PaaS):

This layer is responsible for providing resources such as Operating System and software development frame works. The consumer does not manage or control the underlying cloud infrastructure, network, servers, operating systems, or storage, but the consumer can control the deployed applications and possibly the application hosting environment configurations. Examples of PaaS services are: Force.com, Go Grid Cloud Center, Google AppEngine, AppJet, Etelos, Qrimp, and Windows Azure Platform**.** As shown in figure (1)**.**

Software as a service (SaaS):

This service allows the consumer to use desired software's from the cloud infrastructure. The software applications are accessible from various client devices through a thin client interface such as a web browser (e.g., web-based email).Therefore, consumers are increasingly shifting from locally installed computer programs to on-line software services that offer the same function. Traditional desktop applications such as word processing and spreadsheet can now be accessed as a service in the Web. The consumers have no need to manage or control the cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings. Examples of SaaS cloud service providers are: GoogleApps, Oracle on Demand, SalesForce.com, and SQL Azure etc. As shown in figure (1).

Deployment Models

Public cloud

"cloud made available in a pay-as-you-go manner to the general public". Examples: Amazon Elastic Compute Cloud (EC2), IBM Blue Cloud, Sun Cloud, Google App Engine, Amazon Web Services, and Force.com.

Private cloud

"internal data center of a business or other organization, not made available to the general public."

Community cloud

"shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations)."

Hybrid cloud this fourth class of Cloud infrastructure is a composition of two or more Clouds (private, community, or public). As shown in figure (2).

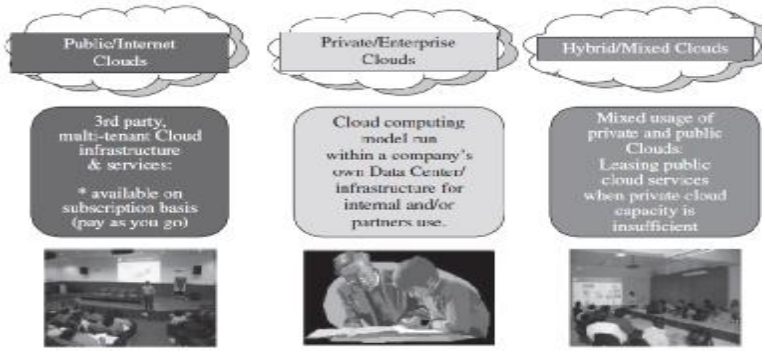


Figure (2) Types of Clouds Based on Deployment Models

Ontology

The word ontology comes from philosophy, where it refers to the study of being or existence. Definition of ontology is so confusing, the most common definition in AIis a method of representing items of knowledge (ideas, facts, things—whatever) in a way that defines the relationships and classifications of concepts within a specified domain of knowledge.

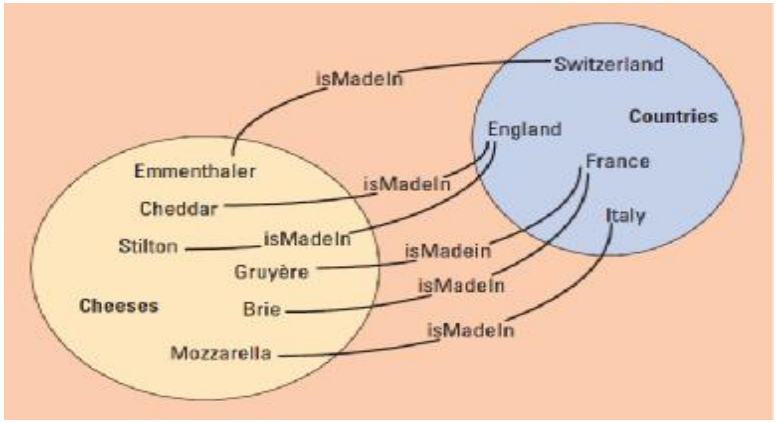


Figure (3)Relationships among classes and Instances illustrate the use of properties

A simple Example to illustrate the concept of ontology, considers a set of cheeses (Brie, cheddar, Emmenthaler, Gruyère, mozzarella, and Stilton). And a set of countries (England, Switzerland, Italy, and France). It's easy to imagine a relationship labeled "is made in" between each cheese and a country. As illustrated in figure (3).

Ontology Building

Ontology building is a process that aims at producing ontology**.** Several methodologies for ontology building have been proposed such as SENSUS, TOVE, METHODOLOGY, ONIONS, ENTERPRISE, commonKADS, PLINIUS, MENELAS, and PHYSSYS.

Ontology Languages

An ontology language is a formal language used to encode the ontology. Ontology languages can be classified as**,** Logical Languages**,** Frame based Languages, and Graph based Languages**.** There are a number of such languages forontologies such as, CycL, Knowledge Interchange Format (KIF), Gellish, RIF and the Web Ontology Language (OWL).The OWL Web Ontology Language is an international standard for encoding and exchanging ontologies and is designed to support the Semantic Web. It was designed to be compatible with the eXtensible Markup Language (XML) as well as other World Wide Web Consortium (W3C)standards; In particular, OWL extends the Resource Description Framework (RDF) and RDF Schema. OWL has three separate flavors: OWL Lite **(**a simple syntax that provides classification capabilities**),** OWL DL **(**a computationally complete version that supports description logic) and OWL Full (a version that provides compatibility with RDF schema).

Multi-agent Systems (MASs)

The term 'agent', or software agent, has found its way into a number of technologies and has been widely used, for example, in artificial intelligence, data bases, operating systems and computer networks literature. There is no single definition of an agent; basic dictionary definition of agent is one who acts. An agent, then, can be a person, a machine, or a piece of software. Agent systems consist of multiple agents. These multi-agent systems (MAS) can model Complex systems and introduce the possibility of agents having common or conflicting goals. These agents may interact with each other both indirectly (by acting on the environment) or directly (via communication and negotiation). As shown in figure (4).

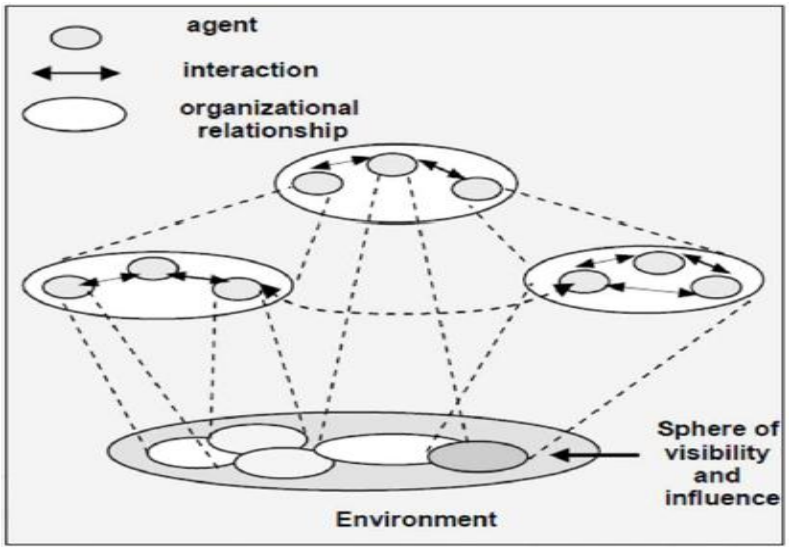


Figure (4) Multi-agent Systems (MASs)

MAS Communication

Agents need to be able to communicate with users, with system resources, and with each other if they need to cooperate, collaborate, and negotiate and so on. In particular, agents interact with each other by using some special communication languages, called agent communication languages; the first agent communication language with a broad uptake was KQML, Currently the most used and studied agent communication language is the FIPA ACL, which incorporates many aspects of KQML.

MAS Negotiation

Negotiation is the communication process of a group of agents in order to reach a mutually accepted agreement on some matter. Negotiation can be competitive or cooperative:

1- Competitive negotiation is used in situations where agents have independent goals that interact with each other; they are not a priori cooperative, share information.

2-Cooperative negotiation is used in situations where agents have a common goal to achieve or a single task to execute. In this case, the multi-agent system has been centrally designed to pursue a single global goal.

MAS Programming Languages

Multi-agent systems can be realized by using any kind of programming language. In particular, object-oriented languages are considered a suitable means because the concept of agent is not too distant from the concept of object. Several agent-oriented languages are available; some take either a purely declarative or a purely imperative programming approach such as (FLUX and JACK Agent Language). Other hybrid languages combine the declarative and imperative features of other languages such as (3APL and Jason).

Proposed System

The proposed system consists of four layers that work simultaneously to achieve the goal of this work. The components of these layers are:

1- Cloud Computing Infrastructure: "Private Oxygen Cloud" is a special software application easy to install in any computer and enable companies to make their own storage accessible across any platform (windows; mac, iPhone, iPad and android) by inter security information (user name and password).

2- Intelligent Multi-agents Platform: This thesis suggested JADE (Java Agent Development Environment) open source software as cloud intelligent multi-agent platform framework allows developers to build applications and services over the internet. It's the most widespread framework in use today and easy extension with add-on modules.

3- Software Agents: Forming a cloud application or service, these agents are autonomous and can interact with other cloud software agents to perform tasks and enable programmer for developing them, some agents has user interface.

4- Deployment Tool: The proposed system used ASCML (Agent Society Configuration Manger and Launcher) as deployment tool to create, configure, launch and monitor cloud society; a cloud society is defined as a set of collaborating cloud agents forming an application or service.

By gathering all the component mentioned before, the complete work will be presented. Figure (5) illustrates the architecture of the proposed system.

Cloud software agents interact with cloud environment using ontological interaction method. The content languages and ontological support for the cloud platform will authorize it to manipulate and deal with information from within the cloud agents in an easy manner in the form of java objects and it will perform the conversion between agents autonomously and will check the ongoing operations with no need for any external interference, exploiting the cloud platform supporting for content language and ontology make cloud agents talk and reason about things and facts related to the domain goes through the number steps, as illustrated in algorithm.

Algorithm

Input: Strings or sequences of bytes

Output: Java objects

Process:

Step1: Defining an ontology including the schemas for the types of predicate, cloud agent action and concept that are pertinent to the addressed domain.

Step2: Developing proper Java classes for all types of predicate, cloud agent action and concept in the ontology.

Step3: Selecting a suitable content language among those directly supported by cloud platform.

Step4: Registering the defined ontology and the selected content language to the cloud agent.

Step5: Creating and handling content expression as Java objects that are instances of the classes developed in step 2

Step6: let cloud platform translate these Java objects to/from strings or sequences of bytes that fit the content slot of Messages.

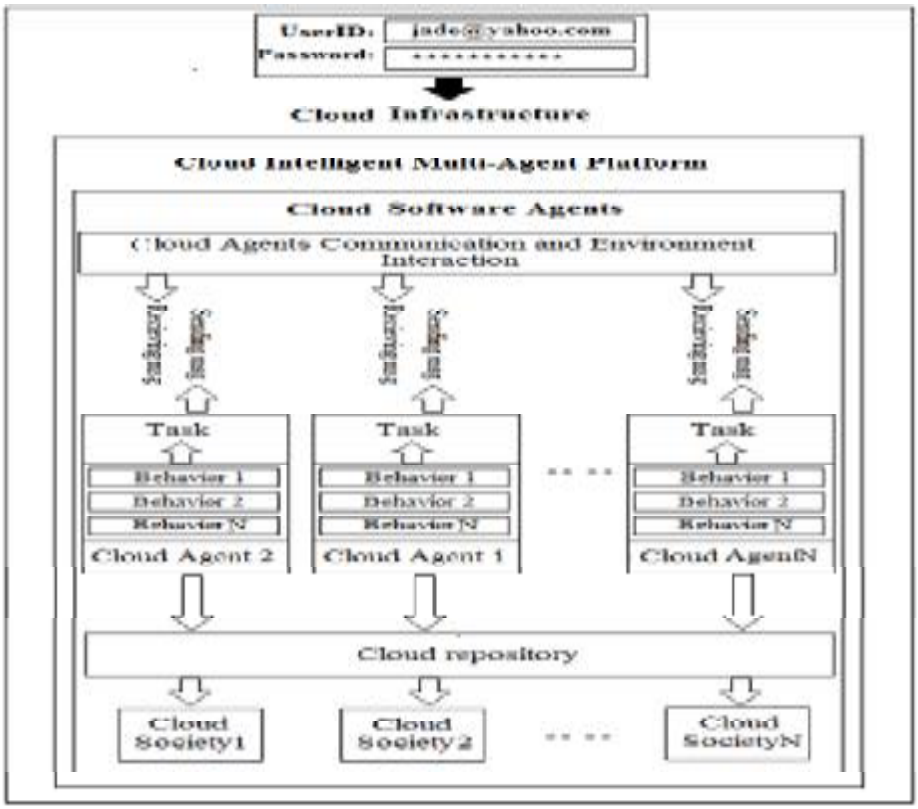


Figure (5) Proposed System Architecture

Proposed System Application
(Ontological Computer Hardware Commerce)

Proposed system application will be used throughout the computer hardware where an agent-based system enables users to commerce computer hardwares. In this application there will be two type of agent (software agent): sellers and buyers. Each buyer agent takes as input some hardware to buy and tries to find agents selling them at an acceptable price. Similarly each seller agent takes as input some hardware to sell and tries to do so at the highest possible price. Both buyers and sellers implement some strategies and carry out negotiation to achieve the best result for the users they represent.

Starting an application consisting of multiple agents usually requires the manual configuration and launch of each agent. This can be difficult, complicated and error-prone. ASCML (Agent Society Configuration manger and launcher) assists in creating different application configurations and manages the launching and monitor of the application, while it is running; the ASCML monitors the run-time states of the agents to check whether the application is operating correctly.

Proposed system Evaluation

Adding intelligence to the cloud environment makes cloud agents perform request of users and takes wise decision with high performance. Information transferred between agents as java objects not require to parse the string each time. This operation reduces response time and allows large number of users to request cloud services (not wait a long time for response). As shown in figures (6,7).

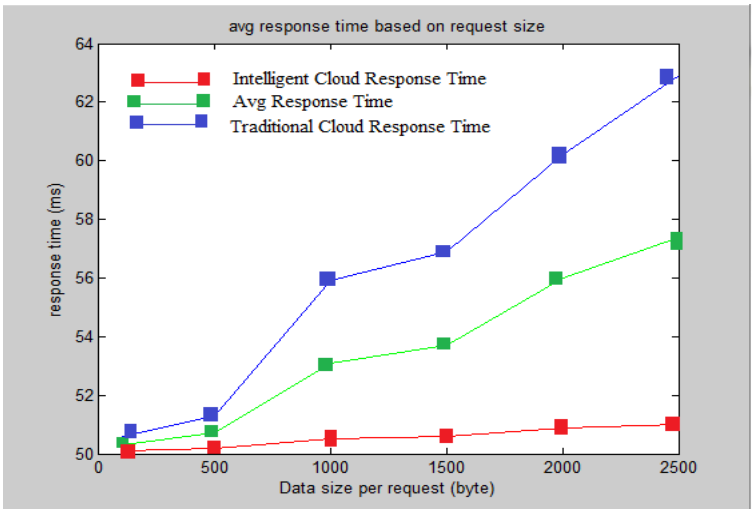


Figure.(6): Average request time

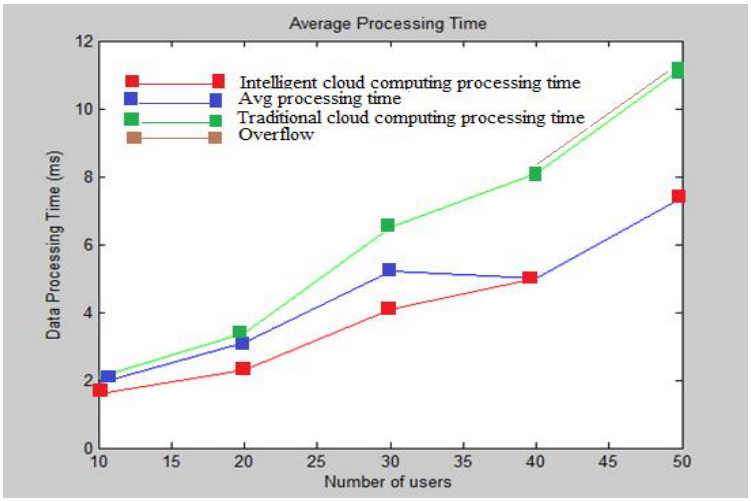


Figure.(7): Average Processing Time

CONCLUSION

In the following points we will display all the conclusions gutted from the implemented proposed system.

1. The proposed system offer cloud computing infrastructure with advance features (powerful, reliable, predictable and scalable) for the execution of multi-agent system implementing complex agent-based applications.

2. The use of Multi-Agent makes the cloud to service in a better way (more flexible, autonomous and capable of intelligent behavior).
3. Software agents interact with cloud environment using ontological interaction for critical thinking and take wise decisions.
3. Using multi agent system in the cloud computing application increases cloud adaptively for any software development.
4. The proposed system increases software sociality by the capability of sharing knowledge and develop new concepts.
5. Applying multi agent system in cloud computing environment increases software internationality by globalizing interfaces to cope with different changes occurred within the environment.

Literature Review

The present survey includes significant previous researches that are related to the thesis objectives:

1.Ronghua Y. and Xiafen Y., [2009]

developed a demo system about multi-agent web services aggregation driven by traveling requirement. In this work, service combines agent to be an active entity, and models services to be intelligent agents which are autonomy, rationality and independence based on ENO (environment ontology) in JADE (Java Agent Development Framework).

2. Taekgyeong H. and Kwang M.S. [2010],

attempted to building an agent-based cloud service discovery system (CSDS) that consults an ontology when retrieving information about cloud services, main contributions of this work is building a Cloud Service Reasoning Agent (CSRA) and cloud ontology to reason about the relations of cloud services using one of reasoning methods and rate the search results, cloud ontology make the CSDS is more successful in finding cloud services that are closer to users' requirements.

3. Aarti S. and Manisha M. [2012],

proposed an agent based framework for providing scalability in cloud computing environments supported with algorithms for searching another cloud when the approachable cloud becomes overloaded and for searching closest data centers with least response time of virtual machine (VM).

Future Work

Some suggestions are given below that could be implemented in the future to make the project more optimal in it's activation with user:

1. Make each agent in cloud takes expert behaviors by using JESS (Java Expert System Shell).
2. Adapting cloud agents in distributed system to move from one cloud to another (mobile cloud agents).
3. Allowing cloud agents to interact with cloud environment using bean ontology that automatically creating all the ontology concepts.

Appendix

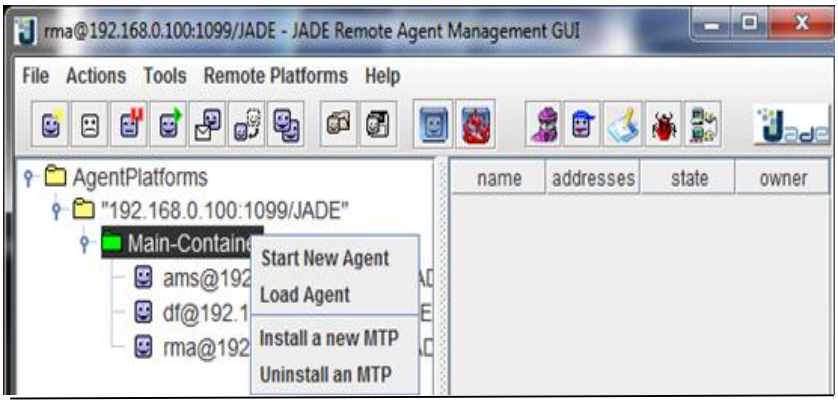
Proposed System Interfaces

- Login private oxygen cloud infrastructure by enter security information (user name and password for security)**, as shown in snapshot (1).**

Snapshot (1)

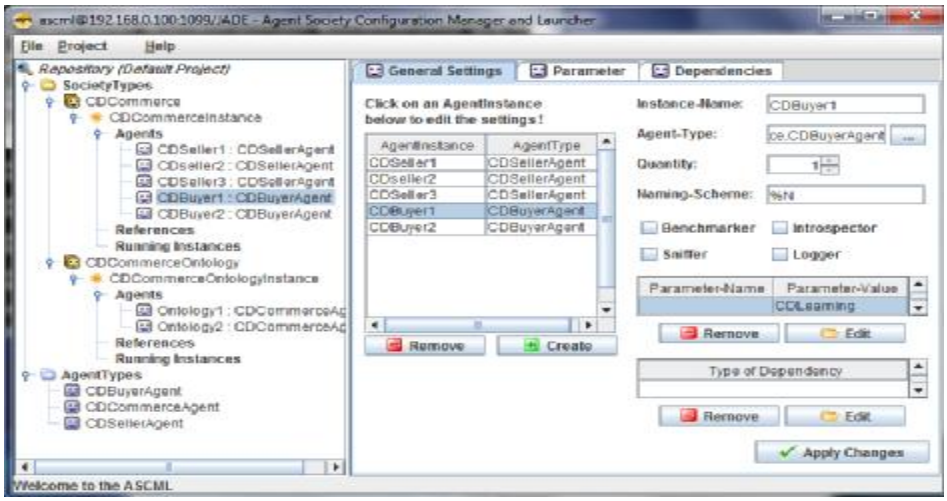
- Oxygen drive (O:\) appears containing cloud platform (JADE) packages and applications source codes,start cloud agents from main container by press "start new agent"**,** as shown in snapshot (2).
-

-



Snapshot (2)

- Then configure and launch application by creating both society and agent descriptions, these descriptions are based on XML meta-model**,** as shown in snapshot (3).



Snapshot (3)

REFERENCES

[1] A. Anasuya T. I., " Cloud Infrastructure Service Management – A Review", IJCSI international journal of computer science issues, Vol.9, issue 2, No 2, March 2012.
[2] Domenico T., "Cloud Computing and Software Agent: Towards Cloud Intelligent services", White paper, ICAR-CNR, 2012.
[3] Rojkumar B., James B., and Andrzej G.," CLOUD COMPUTING: principles and paradigms ", published by John Wiley & Sons in 2011.
[4] Editors: Keith J. [ERCIM], Burkhard N. –lutz [SAP Research], "The Future of Cloud Computing ", Expert Group Report, 2010.
[5] Jon O., "What's Needed for Cloud Computing?", White paper, Enterprise Strategy Group, June, 2010.
[6] Thomas C.J., "Just What Is an Ontology, Anyway?",©IEEE September/October 2009.
[7] B.Chandrasekaran and John R.J., V.Richard B., "What Are Ontologies, and Why Do We Need Them?" © IEEE1999.
[8] Mike U., Scotland & Michael G., "Ontologies: Principles, Methods and Applications", February, 1996.
[9] Jeff H., "AN INTRODUCTION TO THE OWL WEB ONTOLOGY LANGUAGE", 2007.
[10] Michael W., "An Introduction to Multi-agent Systems", UK, 2009.
[11] Fabio B., Giovanni C., Dominic G., "Development Multi-agent Systems with Jade", ©2007.
[12] "Agent Technology ", Green Paper, agent working group, 24 December 1999.
[13] Hyacinth S.N., "Software Agent: An overview", 1996, hyacinth@info.bt.co.uk
[14] Ronghua Y. and Xiafen Y., "Multi-agent Web Services Aggregation Driven by Requirement in JADE", © IEEE, 2009.