Available online at www.qu.edu.iq/journalcm
JOURNAL OF AL-QADISIYAH FOR COMPUTER SCIENCE AND MATHEMATICS
ISSN:2521-3504(online) ISSN:2074-0204(print)

A MULTI-AGENT SYSTEM FRAMEWORK FOR CLOUD RESOURCES ALLOCATION

**** Fouad jowda[a] and Muntasir Al-Asfoor[b]***

[a]*College of Computer Science and IT, University of Al-Qadisiyah, Iraq .Email: com.post10@qu.edu.iq*

[b]*College of Computer Science and IT, University of Al-Qadisiyah, Iraq.Email: muntasir.al-asfoor@qu.edu.iq*

ARTICLE INFO

Article history:
Received: 13 /08/2021
Rrevised form: 25 /08/2021
Accepted : 09 /09/2021
Available online: 20 /09/2021

Keywords:

Cloud computing,

Multi-agent system,

Resource allocation,

MAS.

ABSTRACT

As cloud computing becomes increasingly attractive to entrepreneurs and resource consumers, researchers have recognized that an integrated infrastructure is needed to explore the potential of the cloud and enhance its efficiency and features. The primary people involved in cloud system operations are usually those who use the cloud, search the cloud provider's website, and make the purchase. This challenge is caused by the lack of a mechanism to provide negotiation interfaces through cloud providers to deal with them dynamically. In addition, one of the most common obstacles that consumers face is choosing resources based on their requirements at an affordable cost. Therefore, the goal of the proposed system is to simplify the process of allocating cloud computing resources to the consumer by choosing the most appropriate offer based on the cloud computing architecture and its integration with the Multi-Agent System (MAS) framework. Accordingly, the resulting system is more efficient and responsive, and negotiations can be conducted easily. In order to find mutually appropriate solutions in terms of service quality and price. The proposed system is applicable to all types of cloud deployments. It has been built using Java programming language and uses CloudSim and JADE as the two types of emulators used in the design and implementation of the system.

MSC. 41A25; 41A35; 41A36

DOI : https://doi.org/10.29304/jqcm.2021.13.3.847

1. INTRODUCTION

Nowadays, Cloud Computing has become a common approach for provisioning resources over the internet. It represents the fifth generation of computing after mainframes, personal computers, the client/server paradigm, and the web (World Wide Web). This approach is not entirely new, few years ago, IBM already proposed "on-demand" computing under the systematic ASP (Application Service Provider) approach. It represents the fact of offering an application as a service. The 80s were also the beginning of virtualization technology. All these concepts and

∗ ∗Corresponding author: Fouad jowda.

Email addresses: com.post10@qu.edu.iq.

Communicated by*: Dr. Rana Jumaa Surayh aljanabi.*

technologies have led, little by little, to the invention of a new way of offering IT "as a service". IT services are provided as a utility to describe a model commercial in terms of supply on-demand. In this model, the user pays the service providers based on their consumption. Similarly, when customers obtain any kind of public service, such as electricity, water, phone service, etc.,

Cloud computing is seen as a fully virtualized system, allowing both the calculation, storage, and use of software resources as well as servers as a single platform. Data management services are currently operated in the user's local environment, but they are served remotely by a Cloud Service Provider (CSP). The services are offered in an abstract form. Users may not know where, when, how, why, and by whom their data is viewed or modified in a Cloud environment.

1.1. Cloud Computing

The cloud is a huge virtual resource that can be accessed and used easily [1] [2]. Based on the "pay-as-you-go" concept [3]. The increasing importance of the form has led to a variety of definitions. [4] [5] [6] The most commonly used term, and, in our opinion, the most accurate form both a technological and a realistic standpoint, is given by "NIST" [7]. Mell et Grance suggest this definition. "That Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models".

1.2. Agents

The definition of agent has gained the attention of many researchers for decades in different disciplines. The term agent is a term referred to in general by different entities. It has been used in systems based on biological knowledge, autonomous robots, components, and computer programs, whether integrated into operating systems or complex computer systems, natural language processing, and other areas of artificial intelligence.

There are several definitions of agents in the literature. They all look the same, but they vary based on the application on which the agent was built. With the introduction of emerging technology and the growth of the Internet, however, this idea continues to be utilized in a range of new applications, like resource agents, broker agents, provider agents, interface agents, monitor agents, and so on [8] [9].

Demazeau & Costa [10] define an agent as, "an agent is a real or virtual entity whose behavior is autonomous, evolving in an environment that it is capable of perceive and on which he is able to act and interact with other agents ".

Agents et al [11] proposed the following definition: "A agent is a computer system, located in an environment, and which acts in an autonomous and flexible to achieve the objectives for which it was designed".

1.3. Multi-Agent System

The Multi-Agent System (MAS) emerged as a way to represent scalable dynamic networks [12]. Usually, a system is an organized set of elements contributing to the completion of a given task. By following this definition, we can define the multi-Agent system as an organized group of agents taking responsibility for achieving a common goal. Many different concepts have been suggested, depending on the discipline of research from which they came. The goal is not to list all the definitions, but to choose one that is general and acceptable.

Stone & Veloso [13] A multi-agent is defined as, "A multi-agent system is a loosely coupled network of problem-solving entities (agents) that work together to find answers to problems that are beyond the individual capabilities or knowledge of each entity (agent)."

Jacques [14] "Multi-agent systems implement a set of concepts and techniques allowing heterogeneous software or parts of software, called "agents" to cooperate according to complex modes of interaction They are meant to work together to solve problems that are beyond their ability or knowledge individuals and take advantage of the various roles of each of the agents in the system".

MAS may be utilized to build a far more effective, adaptive, and scalable cloud computing ecosystem architecture than is presently available [12].

1.4. Contributions

A new agent-based cloud computing system has been created, deployed, and used that can filter cloud providers, identify providers that match customer specifications, and conduct negotiations.
This thesis serves as a foundation for cloud computing markets by outlining the strategy that future companies may use to create their cloud computing markets. So that both cloud consumers and cloud service providers can reach each other. There are three advantages for users:

- It will help make the most of the cloud providers' resources by helping to satisfy the needs of prospective cloud customers.
- Cloud users will benefit from this research, because it may assist them in finding an acceptable service and many options that can serve as a baseline for the identification of eligible providers.
- By providing negotiating, it will enable users to renegotiate prices with service providers and get the services they need at the agreed price.

2. Related works

The problems relating to the allotment of resources were outlined by Yann Chevaleyre et. coll. [15]. The major problems are communication protocols, consumer tactics for manufacturers, and algorithms used to allocate the best resources. The protocol may be a central mechanism such as auctions or a distributed such as negotiation for distribution.

Marian Mihailescu and colleagues[16] have developed a dynamic pricing approach. According to the forces of demand and supply, the price of resources is determined according to the market conditions. In their recommendation, a third party known as market makers is brought in to gather bids, choose a winner, and calculate the payout.

Several market-based resource allocation techniques are employed in a multi-agent system to allocate resources. Shneidman et al. [17] describe how markets might be used to address the issue of resource allocation, as well as the difficulties associated with current market-based resource distribution.

Rich Wolski and colleagues [18] describe the methods for implementing a computational grid. They compare the efficiency of resource allocation under two distinct market settings, such as commodities markets and auctions, to determine which is more efficient. Prices, market equilibrium, consumer efficiency, and producer efficiency were all measured and compared between the two market methods.

Resources would be allocated via negotiation with a decommitment penalty, according to Bo An and colleagues If the consumer can reach an arrangement with one of the providers, the other will be required to pay a de-commitment penalty to compensate the consumer. Methods such as the following are used. Consumers must meet their resource needs by the deadline; the seller's cost the provider/seller will not accept a price that is less than their actual cost, and consumers interact with service providers directly via this method.

Xiaominzhu,chao [19] A method was developed in the cloud environment using Agent-based scheduling to assign real-time tasks and to dynamically supply resources. Additionally, to guarantee network quality of service (QoS) and system performance, cloud service providers use job scheduling in real-time.

Haresh and Saidalavi [20] have built the dynamic cloud resources allocation system (deployment and administration of various external and internal cloud computing services to meet the company's requirements). Three agents were created: Consumer agents (CA) (RPA). The primary benefit of this approach is that consumers don't have to worry about where the resource is located and what the cost is to the customer.

Lin and Wang [21] developed a dynamic allocation of resources method that allocates VM dynamically in cloud computing depending on local changes. They used a threshold algorithm to optimize resource reallocation decisions. To find out what resources the cloud application requires, a threshold-based dynamic allocation is implemented. The system will alert us when a higher amount of resources is needed, and we may change the VM volume based on this alert. The experiments on dynamic RA with thresholds, peak load, and resource usage costs were performed

Mazrekaj et.al [22] suggested utility-based dynamic resource allocation (UDA) to get better overall performance in data centers and to increase flexibility by allocating less when the system is loaded but more when it is overloaded. When used, UDA has been shown to improve VM SLA compliance and EVS (Energy and Cumulative SLA Violation).

3. THE PROPOSED SYSTEM'S DESIGN

The proposed framework consists of several agents, as shown in Fig.1. below, where the system is designed to negotiate and dynamically provision resources to the desired user agent. The user does not need to know whom the cloud service provider is and where the resources are located geographically. The consumer always gets the best offers available in cloud computing based on predefined criteria. The system is characterized by flexibility, so that if the number of users increases, the number of agents increases automatically. likewise, when the data center increases for the service providers or decrease it, the agents will increase or decrease according to the situation that occurred.

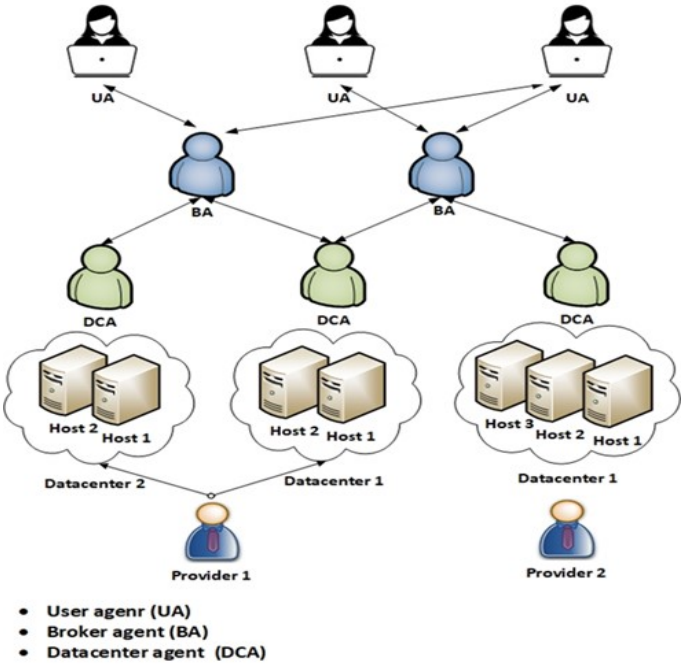


Fig.1. the proposed cloud computing system's design

3.1. User Agent (UA):

The user agent is responsible for receiving requests from users and sending them to the broker's agents through message passing. The results of the request are shown to the user via a user interface provided by the agent. To use the service, the User-Agent communicates with the Provider's Agent through the Broker's Agents. Algorithm 1 of the user agent below is used to perform the work and is as follows.

Algorithm 1: User-agent (UA)
begin

```
1. Register UA in DFA agent                    // DFA is Directory Facilitator agent
2. initialize Price User, Resource User
3. AID [] searchResult = searchDF("BA")        // BA (brokerAgent)
4. While (searchResult == null) do
5. Thread. Sleep (1000)
6. searchResult = searchDF("BA")
7. End While
8. for (AID BA: searchResult)
9. Add receiver (BA) to msg                    // the msg is ACLMessage type REQUEST
10. End for
11. For (int i = 0 ; i < vms.size() ; i++)     // vms A list of the required vm from user
```

```
12. RU =  RU + vms.get(i).getId()          // RU variable type String (getId vm)
13. End for
14. send msg(RU) to BA
15. If receive PROPOSE (cost of RU) from BA  then
16. Evaluate()                             // A function to evaluate the offer received from  BA
17. If PROPOSE () is acceptable then
18. send "ACCEPT_PROPOSAL" to BA
19. else
20. send "REJECT _ PROPOSAL"
21. goto step 6
22. End if
23. End if
End
```

3.2. Broker agents (BA):

It receives a request from the user agent, then sends a CFP () to the data center agents to start the bidding round. The data center agents receive offers from the data centers after a specified time, evaluate the offers received from the data centers, and select the best bid submitted, then send that bid to the user agent. He receives a response from him. By agreeing to the offer, the agent will be asked to carry out the tasks assigned to him or refuse the offer. In this case, the broker's agent will conduct a negotiation process between the data center agent and the user agent, and the negotiations will be successful or unsuccessful depending on what is involved in the negotiation process between the agents. Algorithm 2 of the Broker agent below is used to perform the work of this agent and is as follows.

Algorithm 2: Broker agent (BA)

```
begin
1.  Register BA in DFA agent                  // DFA is Directory Facilitator agent
2.  If receive REQUEST (RU) from UA then
3.  AID[] searchResult = searchDF("DCA")      // DCA (datacenteragent)
4.  While (searchResult== null) do
5.  Thread. Sleep (1000)
6.  searchResult = searchDF("DCA")
7.  End While
8.  For (AID DCA: searchResult)
9.  Add receiver (DCA)  to msg                // the msg is ACLMessage  type CFP
10. End for
11. send msg (RU) to DCA
12. End If
13. If handle Refuse from DCA then
14. Print "Agent Refuse"
15. end If
16. If handle failure from DCA then
17. If no replay then
18. Print "Reply does not exist"
19. else
20. Print "name Agent sender failed"
21. End If
22. End If
23. If  handle All Responses from DCA  then
24. save All Responses in e as Enumeration
25. While (e. has more elements()) do
26. Calculate the cost of the VM depending on the bid by eq (1)
27. evaluate PROPOSE ()
28. If PROPOSE () is acceptable then
```

29. send “ACCEPT_PROPOSAL” to DCA
30. **else**
31. send “RECJECT _ PROPOSAL”
32. **end If**
33. **end While**
34. **If** handle Inform from DCA **then**
35. send the best PROPOSE() to UA
36. **End If**
37. **If** receive “ACCEPT_PROPOSAL” from UA **then**
38. send REQUEST() to PlataformAgent
39. **End If**
40. **If** receive “ REJECT_PROPOSAL “ **then**
41. send “REFUSE” to DC
42. **End If**
43. **If** receive “CONFIRM” from DCA **then** // Receipt of a proposal after the negotiation process
44. send PROPOSE (Cost after Reduce) to UA
45. goto 8
46. **End If**
47. **If** receive “REFUSE” from DCA **then**
48. **If** (the number of CFP <DCA) **then**
49. goto 3
50. **else**
51. send “FAILURE” to platform agent
52. **End If**
53. **End If**

End

3.3. Datacenter agents (DCA):

Each service provider owns one or more data centers. At each data center location, there is a data center agent who makes proposals on behalf of each data center and negotiates with the user agent to provide services, the algorithm for the work of the data center agent is algorithm 3, as shown below.

Algorithm 3: Datacenter agent (DCA)

begin

1. Register DCA in DFA agent
2. initialize (number of time negotiation, discount percentage) // for each DCA
3. **If** receive CPF(RU) from BA **then**
4. Matching Protocol FIPA-CONTRACT-NET
5. Prepare Response ()
6. send PROPOSE **(**Cost of RU**)** to BA
7. **End if**
8. **If** receive “ACCEPT PROPOSAL” from BA **then**
9. send “INFORM” to BA
10. Print “DCA Proposal accept “
11. **elseif** receive “REJECT_PROPOSAL” from BA **then**
12. Print “DC Proposal rejected “
13. **End if**
14. **If** receive “REFUSE” from BA **then**
15. **If** it is possible to negotiate **then**
16. Reduce cost of RU
17. send “CONFIRM” and new price to BA
18. **else**
19. send “REFUSE” to BA

20. **End if**
21. **else**
22. Block ()
23. **End if**

end

3.4. Platform Agent (PA):

This agent has the primary task of implementing or canceling the implementation of resource allocation where the simulation process is resumed if the offer submitted by the service provider has been approved by the user's agent and the execution is canceled if the process ends with the allocation of resources in failure after choosing the best offers and conducting negotiations with all service providers. The algorithm for the work of the Platform agent is algorithm 4, as shown below.

Algorithm 4: Platform agent (PA)

begin

1. Register PA in DFA agent
2. **If** receive REQUEST (carry out tasks**)** form BA **then**
3. send "AGREE" to BA
4. Execution (task)
5. **else**
6. send "NOT_ UNDERSTOOD" to BA
7. **End if**
8. **If** receive "FAILURE "from BA **then**
9. Stop_ Simulation ()
10. **End if**

End

4. The Proposed Model Diagrams

The whole organization of agents is visualized using the integrated interaction diagram, which displays the organizational structure as it is used in the problem domain. Fig.2. shows the agents and how they interact.

The broker is the core behind the cloud infrastructure. It can communicate with agents. The broker acts as an agent manager in and of itself, coordinating communications between clients, monitoring running tasks, and keeping clients' agents informed about the operation. In this paper, the broker has moved from preserving the specifications of the cloud service providers to becoming an agent for processing results, in addition to recruiting clients and finding the best service providers suitable for the consumer.

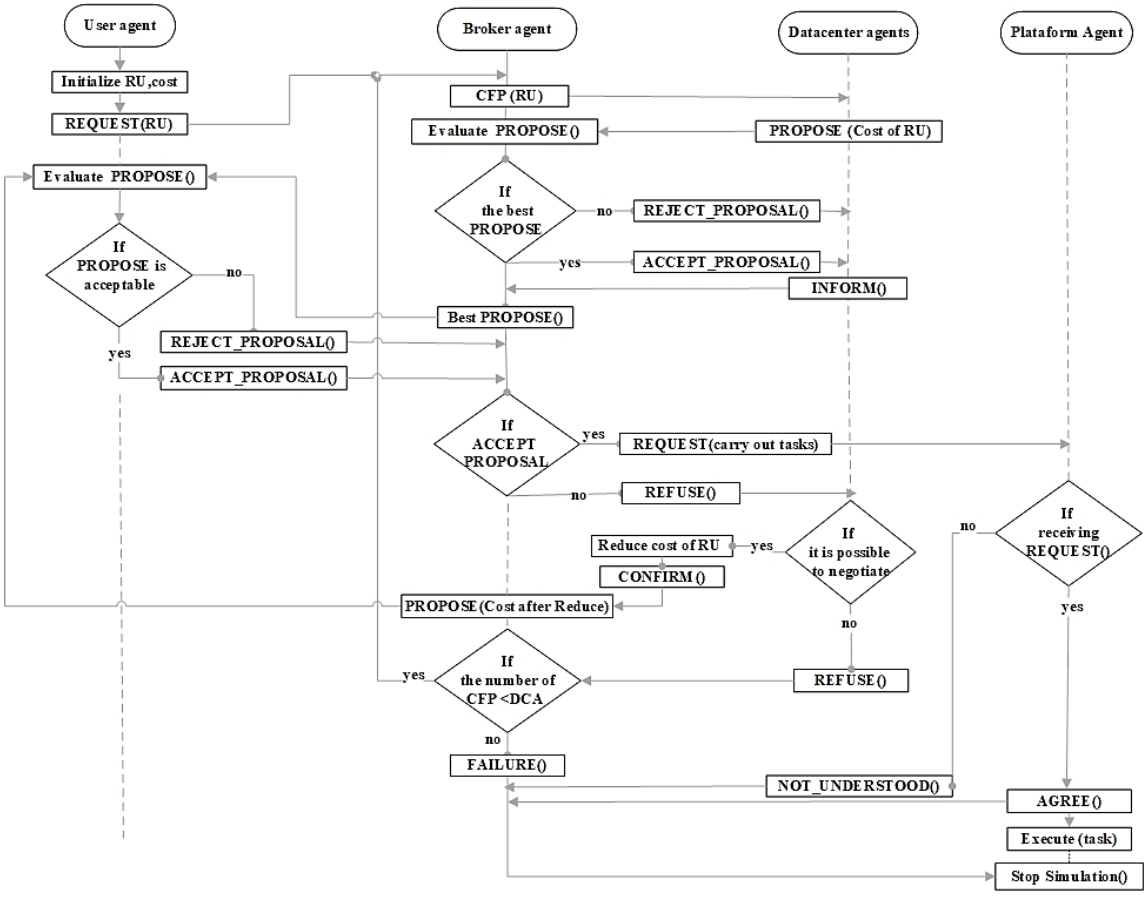


Fig.2. The proposed model diagrams

The user agent performs the work according to algorithm 1 as it initializes the request message and sends it to the broker agent. After receiving the request, the broker agent extracts the required resources from the request and searches for all data center agents, and sends it to the "CFP" forbidding and waits for a full set of responses (or timeouts).

The broker's agent performs the work according to algorithm 2 as it evaluates the offers received after Calculate the total cost of the resource set of each offer presented as in equation No. 1, making sure that the previously submitted offer will be excluded from the competition between offers. This process is done by checking the data of the service providers and their offers registered with the broker's agent who chooses the most appropriate offer. Then it sends the offer acceptance and waits for a response "INFORM" from the data center agent whose performance is according to algorithm 3. The offer is then sent to the user agent through the "PROPOSE" message, who evaluates the offer and responds by accepting or rejecting the offer. The user agent sends the "ACCEPT_PROPOSAL" to the broker agent if the cost is acceptable. If the cost is higher than expected, send it "REJECT_PROPOSAL"

$$\text{Total\_cost}= \sum_{1}^{n}(q(\text{ram}) * \text{Costram}) + q(\text{storge}) * \text{Coststorge}) + (q(\text{cpu}) * \text{costcpu}) + (q(B) * \text{costB}) \quad (1)$$

where:

n: the number of VM ordered by the user.
q: a quantity of resources that are required by the user.

If the broker's agent receives the "ACCEPT_PROPOSAL", a REQUEST (carry out tasks) is sent to the platform agent. In the event of a "REJECT_PROPOSAL", the resource data center agent fails to meet a Consumer Agent's request, the Broker Agent opens a negotiation with the resource data center agent. The terms of the agreement or negotiation process must be well understood by both parties (agents) to effectively finalize the agreement and all that. Willing

to achieve the conditions presented for the contract period in a simple net contract protocol, the parties are legally bound by the terms of the agreement. Other agents who bid on the contract may have previously taken on other commitments and are no longer willing to accept the offer from the agent who is no longer able to achieve the contract. This means that it is expected that the entire procedure will need to send a new request.

5. Simulation and Experimental Results

5.1. Simulation Environment

The proposed system is primarily based on the JADE ("Java Agent Development Framework"). It is a platform for creating, controlling, and developing multi-agent systems written in the Java programming language, by F. Bellifemine and A. Poggy, G. Rimassa, P. Turci for the company CSELT (Italy) in 1999. JADE adheres to the 1997 FIPA Standards ("Foundation for Intelligent Physical Agents"). The platform provides generic reception, identification, and communication services between agents.

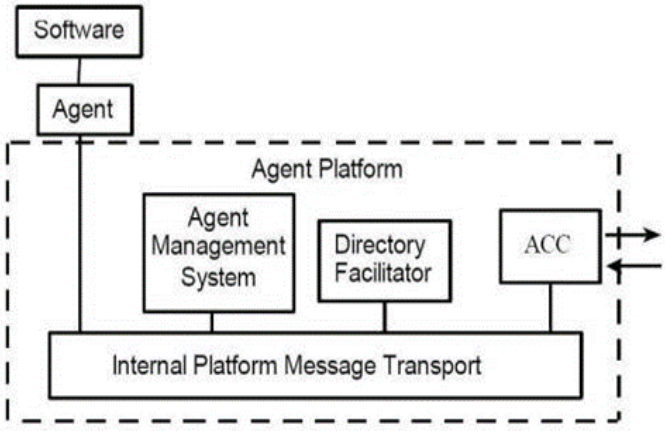


Fig. 3. FIPA Reference model for Agen [23]

The simulation is performed in Fig.4. showing the appearance of the JADE platform used throughout the testing phase.

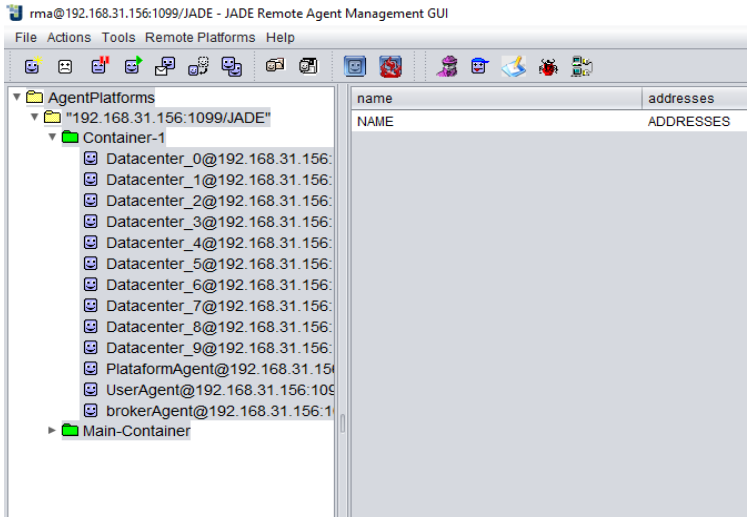


Fig.4. The Simulator Platform

5.2. Simulation Settings

The system consisting of several agents is simulated according to Table 1- below for each agent with a specific task.

Table 1- The number of agents during the test

No	Name agent	Number agent
1	User agent	100
2	Broker agent	1
3	Datacenter agent	10
4	Platform agent	1

Assumptions and conditions for the scenario: The use of two virtual machines is detailed in the Table 2- below.

Table 2- Hardware specifications for VM

Feature	VM
Storge	10 GB
RAM	512 MB
MIPS	250
Bandwidth	1000 MHz
Number of Pes	1
VMM	XEN

Second, using the number of hosts as per the specification in the Table 3-

Table 3- Specifications of Hosts

Feature	Host 1	Host 2
Number of Pes	4	2
MIPS	10000	10000
RAM	100 GB	100 GB
Bandwidth	10000 MHz	10000 MHz
Storage	10 TB	10 TB

Third, the number of hosts (20 hosts) is determined; Equally divided into ten groups, each data center has two hosts. Fourth, the characteristics of the amount of VM required are the same and the cost required for each type of virtual machine is different because each service provider has its price as in the Table 4- below shows the prices set by the data centers, In addition to the number of negotiation times and the percentage of reduction for each data center**.**

Table 4- Data Center Information (Price and Negotiation)

Name DC	Cost cpu	Cost ram	Cost storge	Cost Bw	Num times negotiate	Discount percentage
Datacenter_0	4	0.33	0.02	0.01	3	2
Datacenter_1	4	0.372	0.02	0.01	3	3
Datacenter_2	5	0.51	0.01	0.01	2	1
Datacenter_3	6	0.51	0.01	0.02	1	5
Datacenter_4	5	0.59	0.01	0.01	0	0
Datacenter_5	4	0.4	0.02	0.02	2	1
Datacenter_6	5	0.5	0.01	0.02	0	0
Datacenter_7	4	0.43	0.02	0.01	0	0
Datacenter_8	5	0.381	0.02	0.03	2	4
Datacenter_9	6	0.5	0.01	0.01	1	1

5.3. Experimental results

In this section, experimental results will be presented. The test has 100 users, and each user has a price to pay for buying resources from cloud computing, and each of them has two tasks to perform using those resources. Consumer prices are read from the CSV type file to be constant for all models. About the resource cost prices that will be presented as offers to the user, each data center has its prices as in Table 4-

The proposed model was evaluated based on several factors that reflect the research objectives and compared with other models. These factors include the number of execution and failure cases, and the second factor is the number of messages exchanged in the system, which reflects the system's effectiveness and success in allocating computing resources using the multi-agent system.

The first model test. which represents the standard model of the agent using the CNP interactive protocol between agents by exchanging messages, the second model is a negotiation algorithm between agents to allocate the required resources to the user with the interaction between agents using CPN, the third (proposed) model. This model re-requests the submission of offers from the data center agents through the intermediary agent in the event of not reaching an agreement that satisfies the user agent and the data center agent. The stages of implementing the model as shown in Fig.2.

The system execution state is tested by determining the number of successes and failures. The test was repeated using more than one paradigm in each experiment.

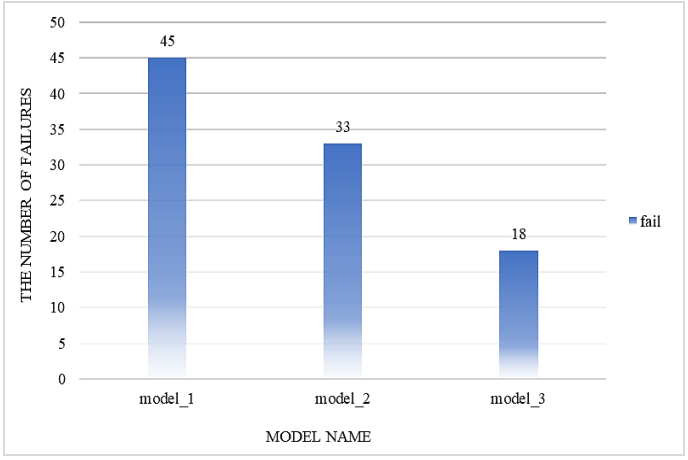


Fig.5. users' request execution failures

Fig.5. represents a comparison of the three models in terms of failure. We note that the highest failure rate was the first model that could not meet the needs of users, where the failure rate was 45%, while the second model was 33%, and the last model, which is the proposed model, we note a very low failure rate of 18%, meaning that the system could not meet the consumer's needs. Only 18 users out of the total number of users (100 users)

The Fig.6. below represents a comparison of the success cases of the three models, including the proposed one. We note in the figure that the third (proposed) model contains the highest percentage of the first and second models, and this indicates the success and efficiency of the proposed model.

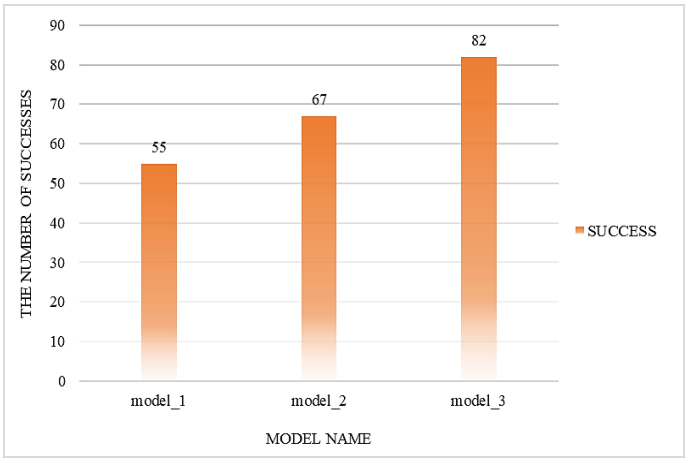
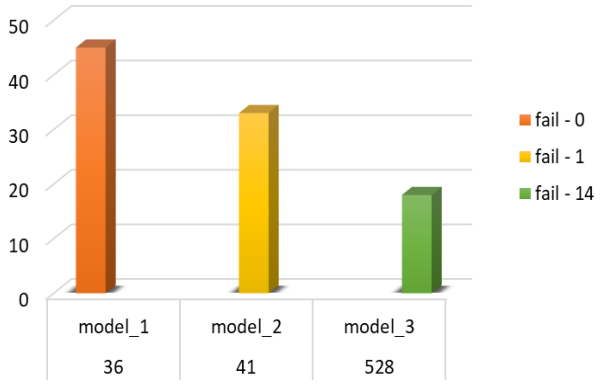
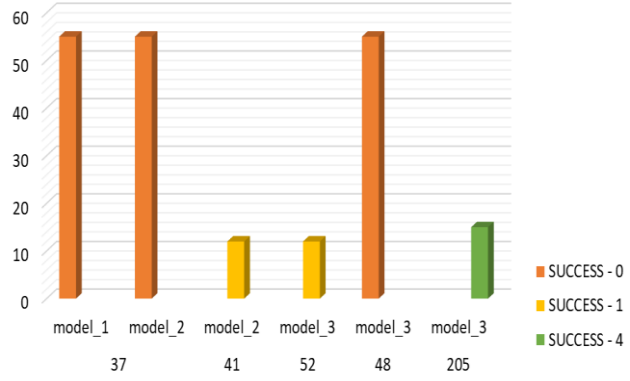


Fig.6. the success of the implementation of users' request





The second test of the proposed system is through the number of messages exchanged in each form.

Fig .7. the number of messages exchanged in case of failure

Fig .8. the number of messages exchanged in case of success

Through the above numbers, we note that Fig.7. and Fig.8. show that the third model contains the largest number of messages in cases of success and failure, and this indicates that the system conducted the negotiation process

more than once and with more than one data center agent. From here, we conclude that the system is more efficient and successful than other models because it negotiates with data center agents in search of a suitable offer for the consumer.

6. Conclusions and Future Works

Cloud computing has emerged as a new computing paradigm to provide a reliable and dynamic on demand IT infrastructure without incurring large amounts of money or heavy administrative responsibilities. If there is a need for a huge and growing supply of computing resources in the cloud, then the cloud computing system is designed to handle resources on a scale that normal computer systems cannot handle. Multi-agent System (MAS) provides a flexible, demand-driven and intelligent approach for Cloud resources allocation. During the course of this research, a new approach for designing cloud resource provisioning has been introduced. The foundation of this approach is to inject MAS within the cloud infrastructure. Therefore, the system performs tasks instead of service providers and the users as it chooses appropriate offers that fit the user's requirements. More after a negotiation strategy between users and providers has been implemented to make to enhance the provided services.

References

[1] M. Armbrust *et al.*, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[2] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Futur. Gener. Comput. Syst.*, vol. 25, no. 6, pp. 599–616, 2009.

[3] A. D. JoSEP, Ra. KAtz, A. KonWinSKi, L. E. E. Gunho, Dav. PAttERSon, and Ar. RABKin, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[4] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *2008 grid computing environments workshop*, 2008, pp. 1–10.

[5] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the clouds: towards a cloud definition." ACM New York, NY, USA, 2008.

[6] L. Wang *et al.*, "Cloud computing: a perspective study," *New Gener. Comput.*, vol. 28, no. 2, pp. 137–146, 2010.

[7] P. Mell and T. Grance, "The NIST definition of cloud computing," 2011.

[8] F. Ackermann, L. A. Franco, B. Gallupe, and M. Parent, "GSS for Multi-Organizational Collaboration: Reflections on Process and Content," *Gr. Decis. Negot.*, vol. 14, no. 4, pp. 307–331, 2005, doi: 10.1007/s10726-005-0317-4.

[9] I. Jarras and B. Chaib-Draa, "Overview of multiagent systems," Cirano, 2002.

[10] Y. Demazeau and A. Costa, "Populations and organizations in open multi-agent systems," *… 1st Natl. Symp. …*, no. June, pp. 1–13, 1996, [Online]. Available: http://www.rocha.c3.furg.br/arquivos/download/demazeau-costa-populations-organizations.pdf.

[11] A. Agents, M. Systems, and Q. Mary, "A Roadmap of Agent Research and Development," *Auton. Agent. Multi. Agent. Syst.*, vol. 38, no. 1.1, pp. 7–38, 1998.

[12] M. Al-Asfoor and M. Fasli, "A study of multi agent based resource search algorithms," *2012 4th Comput. Sci. Electron. Eng. Conf. CEEC 2012 - Conf. Proc.*, no. September, pp. 65–70, 2012, doi: 10.1109/CEEC.2012.6375380.

[13] P. Stone and M. Veloso, "Multiagent Systems : A Survey from a Machine Learning Perspective 1 Introduction 2 Multiagent Systems," *Auton. Robots*, vol. 8, no. 3, pp. 345–383, 1997.

[14] F. Jacques, "Multi-agent systems. Towards collective intelligence," *InterEditions, Paris*, vol. 322, 1995.

[15] Y. Chevaleyre *et al.*, "Issues in Multiagent Resource Allocation," *Informatica*, vol. 30, pp. 3–31, 2006.

[16] M. Mihailescu and Y. M. Teo, "Strategy-proof dynamic resource pricing of multiple resource types on federated clouds," in *International Conference on Algorithms and Architectures for Parallel Processing*, 2010, pp. 337–350.

[17] J. Shneidman *et al.*, "Why Markets Could (But Don't Currently) Solve Resource Allocation Problems in Systems.," 2005.

[18] R. Wolski, J. S. Plank, J. Brevik, and T. Bryan, "Analyzing market-based resource allocation strategies for the computational grid," *Int. J. High Perform. Comput. Appl.*, vol. 15, no. 3, pp. 258–281, 2001.

[19] X. Zhu, C. Chen, L. T. Yang, and Y. Xiang, "ANGEL: Agent-based scheduling for real-time tasks in virtualized clouds," *IEEE Trans. Comput.*, vol. 64, no. 12, pp. 3389–3403, 2015.

[20] M. V Haresh, S. Kalady, and V. K. Govindan, "Agent based dynamic resource allocation on federated clouds," in *2011 IEEE Recent Advances in Intelligent Computational Systems*, 2011, pp. 111–114.

[21] W. Lin, J. Z. Wang, C. Liang, and D. Qi, "A threshold-based dynamic resource allocation scheme for cloud computing," *Procedia Eng.*, vol. 23, pp. 695–703, 2011.

[22] S. P. Rajan, "A Significant and Vital Glance at 'Stress and Fitness Monitoring Embedded on a Modern Telematics Platform,'" *Telemed. e-Health*, vol. 20, no. 8, pp. 757–758, 2014.

[23] Https://web.fe.up.pt, "https://web.fe.up.pt/~eol/SOCRATES/Palzer/multiagentsystems.htm, 'Multi-agent systems,'" *https://web.fe.up.pt*.

[24] Jaber Jawad Al-Asfoor, M. (2017). A Simulation of a Networked Video Monitoring System Using NS2. Journal of Al-Qadisiyah for Computer Science and Mathematics, 9(1), 117-131. Retrieved from https://qu.edu.iq/journalcm/index.php/journalcm/article/view/21.

[25] Adil Raheem, O., & Saleh Alomari, E. (2018). An Adaptive Intrusion Detection System by using Decision Tree. Journal of Al-Qadisiyah for Computer Science and Mathematics, 10(2), Comp Page 88 - 96. https://doi.org/10.29304/jqcm.2018.10.2.387.

[26] Sabeeh Mahmood, G., Mohammed Hasan, T., & Mudheher Badr, A. (2017). Multi-Authority System based Personal Health Record in Cloud Computing. Journal of Al-Qadisiyah for Computer Science and Mathematics, 9(1), 108-116. Retrieved from https://qu.edu.iq/journalcm/index.php/journalcm/article/view/20.