DOI: http://dx.doi.org/10.26483/ijarcs.v9i5.6289

Volume 9, No. 5, September-October 2018

ISSN No. 0976-5697

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

ASSESSING PERFORMANCE OF AGENTS IN A MAS: AN EXPERIMENTAL STUDY

S.Ajitha, T. V. Suresh Kumar
Ramaiah Institute of Technology,
Bangalore-54, India

Abstract: A Multi-Agent System (MAS) is understood as a system consisting of interacting autonomous agents.Research on MASis mainly concerned with functional properties such as coordination, rationality, and knowledge modeling. A very less attention is paid for the research activity on the nonfunctional properties performance, scalability and reliability of the MAS. However, as the MAS technologies have gradually matured to be exploited in building practical distributed applications, the non-functional properties have become increasingly important, and it is now vital to pay attention to the issues of nonfunctional characteristics of agents. We propose to access the performance of agents in MASand the system is implemented using JADE. The experimental study provides the opportunity for cause and effect relationships. We considered a tiered architecture for the MAS. A tiered design can lessen the overall impact of changes to the application and allows modifying a component without disturbing the next Tier. So with the help of a good deployment plan, we can distribute the layers over multiple physical tiers in distributed computingto implement a better architecture of the system. The case study considered for illustration is Supply-Chain Management and the results obtained are validated with the tool SMTQA.

Keywords: Multi-Agent System, Performance, JADE, Tired Architecture, Supply Chain Management, SMTQA.

1. INTRODUCTION

A Multi-Agent System (MAS) is generally understood as a system comprised of interacting autonomous agents. Multi-Agent System is relatively a new paradigm in the field of computer science. This paradigm proposes solutions to highly distributed problems in self-motivated, open computational domains. Research on multi-agent systems has been mainly focused on functional properties such as coordination, rationality, and knowledge modeling. A very less attention is paid for the research activity on the nonfunctional properties such as performance, scalability and reliability of the MAS. However, as the MAS technologies have gradually matured in building many practical distributed applications, the research activities towards the non-functional properties of the MAS have become important, and it is now vital to pay attention to the issues of nonfunctional characteristics of agents.

Software Performance Engineering (SPE) is a process to predict the performance of software systems early (analysis phase) in the life cycle and to monitor, report actual performance against specifications and predictions [1,2]. From the software point of view, the process is based on the availability of software artifacts that describe the suitable abstraction of the final software system. Since performance is measured at run time, performance analysis requires appropriate descriptions of the software run time behavior. It may be alternatively referred to as software performance engineering within software engineering.

Determining performanceafter the development of the software is a common industrial practice, but this will lead to the usage ofcostlier, and powerful hardware than what was originally anticipated. This leads to a rigorous tuning measures, or redesigning the complete application. The discovery of unsatisfactory performance, at the later stage of can cause a costlier redesign and implementation of the software or hardware and this leads to the delay in delivery of the system. To overcome such situations, the system's performance characteristics must be considered throughout the development of the software. The goal of SPE is to guarantee that a software system will meet its performancegoals before the system is implemented.

2. RELATED WORK

This section gives brief information about the research activities carried out by different researchers in the field of MAS. The authors in [3], discussed the different research and development activities in the field of autonomous agents and Multi-Agent Systems. They identified the key concepts and applications, and to indicated how they relate to one-another. Some of the historical context of the field of agent-based computing, contemporary research directions are presented and a range of open issues and future challenges are highlighted. In [4], the authors presented a survey of Multi-Agent Systems (MAS) which gives an insight to the field of agents and the organizational framework of agent systems. A number of general Multi-Agent scenarios are also presented by the researchers. The issues that arise for different scenario are discussed along with the different techniques that exist to build Multi-Agent Systems. The presented techniques are not exhaustive, but they highlight how Multi-Agent concepts can be used to build complex systems.

Agents are autonomous and can operate in open electronic environments that are now becoming very popular. Agent technology allows software engineers to develop solutions, which can co-exist and operate along with external and legacy systems. This is important in the context of SCM that involves many parties, which might use different technologies. The Multi-Agent approach is "a natural way to modularize complex systems" [5] which can be easily adopted for SCM.This method allows splitting different tasks within the SCM and executing them both independently and in coordination with each other. The whole software can be made into separate building blocks; each module can be focusedto a particular part of the supply chain. By replacing one building block with another and by combining them in different ways, different versions of the system can be developed and analyzed. In this way, the influence of changes in behavior in each link of the supply chain can be thoroughly analyzed.

MAS is becoming a currentapproach for modeling and developing complex systems such as supply chains. Presently, there exist no standard methodologies for modeling supply chains using MAS. A generic process-centered methodological framework, Multi-Agent Supply Chain Framework is proposed to (MASCF) to simplify MAS development for supply chain (SC) applications in [6]. Depending on the specific roles of software

agents, a new specific interaction pattern is discussed and implemented. An emergency control approach to prevent the wide spread power interruption using MAS is addressed in [7]. The control algorithm was based on decentralized architecture of intelligent agents to achieve fast and accurate response when a catastrophic disturbance is identified in the system.

The SPE approach proposed by Connie U. Smith was the first methodology to the integration of software performance analysis into the software engineering process [1,2]. The SPE process requires additional data that includes software resource requirements for processing steps and computer configuration data. The analysis of the software model gives information about the resource requirements of the software system. The obtained results, together with information about the execution environment, are the input parameters of the system execution model. The analysis of system model helps to find out the point of resource contention, sensitivity of performance metrics to variations in workload composition, service level objectives, and identification of bottleneck resources. A process to elaborate the analysis results and to score performance requirements, model entities and guilty performance anti-patterns is introduced by Cortellessa et al [8].For quantitative evaluation of the performance of the software systems an approach is proposed by Balsamo et al [9,10]. They developed a prototype tool for automatic translation of the software model into a process-oriented simulation model.

Load balancing issues [11], with reference to agent properties and load balancing techniques and the space of load-balancing design choices in the arena of multi-agent computing is explained in detail. In view of the special agent characteristics, a communication-based load-balancing algorithm is proposed, implemented, and evaluated. This algorithm works by associating a credit value with each agent. For an agent based system development, [12] an analytical approach for performance improvement is discussed. This approach avoids the need for a prototype implementation since architects can determine the overall form of the performance equation from the architectural design description and can enhance the system architecture to derive optimal architecture from the analytical model.A framework and a number of negotiation performatives[13], which can be used to construct pair wise and third party negotiation protocols for functional agent working is explained by the authors. They also explained how to formally model the negotiation process by using Colored Petri Nets (CPN) and provided an example of establishing a virtual chain by solving a distributed constraint satisfaction problem. The authors presented a methodology for the assessing the performance of the distributed multi agent system [14]. They considered the characteristics of the system and spotted performance metrics.In [15] the authors developed a supply chain model that uses agent communication, coordination and negotiation between the agents to achieve the intended business goals. They used Java Agent Development Framework (JADE) to implement the supply chain environment. In [16] the authors outline a Collaborative Material Procurement System architecture, which automatically retrieve software based services for the agents that coordinate the supply chain form a service repository.

3. ARCHITECTURES CONSIDERED FOR THE STUDY

The agent technology has become the most popular tool for designing distributed applications. SupplyChain Management (SCM) systems is a best choice as it provides an adaptable and dynamic way for managing separate links within the chain. Unlike centralized approaches, agent-based SCM systems can respond quickly to changes and disturbances (either internal or external) through local decision making. Another advantage of designing the SCM solution as a Multi-Agent System (MAS) is that it allows different tasks within the SCM to be separated and explored both independently and in relation to each other. The case study we considered for the implementation of the SCM consists of five agents namely Manager Agent (MA), Production Agent (PA), Inventory Agent (IA), Supply Agent (SA) and Delivery Agent (DA). We have implemented the model using JADE environment. The results are obtained by considering different tiers.

Performance of a software system is closely tied to the software architecture it follows. It is advantageous to analyze the performance of software from its architecture as such an analysis can be done early in its development. The software architecture determines the way the different components that make the software, interact with each other. Moreover, it also defines the deployment or the arrangement of the components in the available hardware. Thus, while analyzing the performance of any software system taking its architecture into account becomes very important. It is advisable to evaluate a software, early in its development for performance attributes such as the response time,

throughput, etc. This allows the designer to ascertain how the system will perform once it is ready. Moreover, the behavior of the system with regards to changes in the workload, such as an increase in the number of users on system performance is also useful to know while designing the system. Such evaluation is helpful for existing systems, to ascertain their performance behavior under different workloads. Keeping in view of this, we conducted a study on three different types of architectures.

3.1 Single Tier Architecture

We first implemented our case study as single tier architecture. We have considered five agents for our application. The agents considered are Manager Agent, Production Agent, Inventory Agent, Supplier Agent and Delivery Agent. Fig1presents the set up, we considered for the single tier architecture.

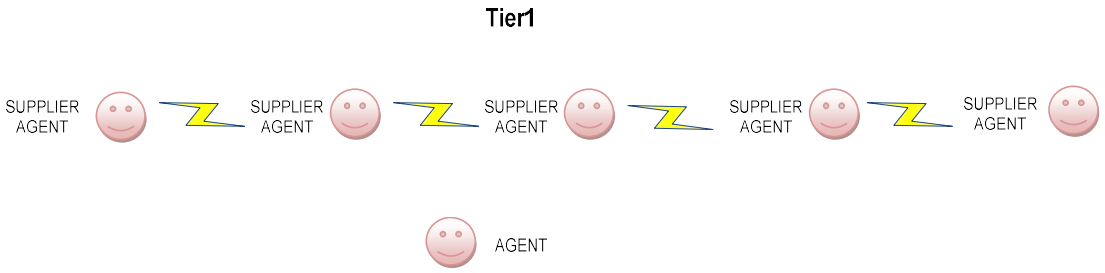

Fig.1. Representation of single tier architecture

3.2 Two Tier Architecture

Next we implemented the case study by considering two tier architecture. We have considered a total of seven agents for implementation. In tier one only Interface Agent is considered. In the second tier, we have considered the Sales Agent, Factory Agent, Inventory Agent and Supplier Agent. In this setup, we considered one agent as a Manager agent for the remaining agents in the second tier. The scenario we considered is presented in Fig2.

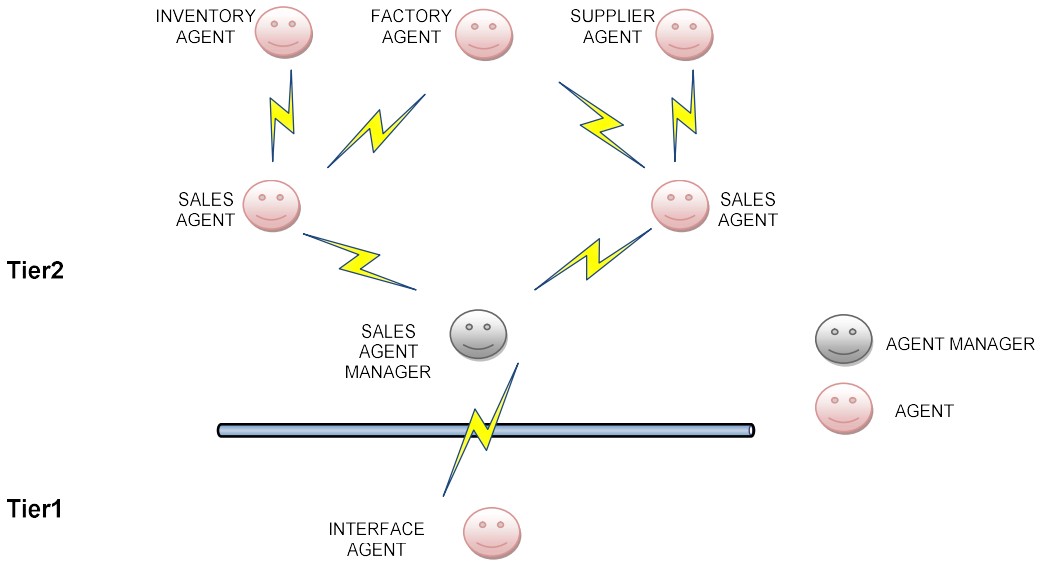

Fig.2.Representation of two tier architecture

3.3 Three Tier Architecture

Next we implemented the case study by considering the three tier architecture. Three tiered architectures fully insulate clients from business rules, the underlying data storage, and concurrency issues, resulting in complete encapsulation. Because clients only interact with three tiered architectures, changes can be made in the database without having to touch a single line of code at the client. Applications have an open architecture and are fully scalable. Applications built with three tiered architectures is also much more maintainable. The architecture separates responsibility into loosely coupled layers (i.e., user interface, three-tiered architecture, and data storage). Because of this separation of responsibility applications can be modified more easily when the business needs change. All business logic and data storage code reside on other machines across the network.

We have considered a total of thirteen agents. In tier one, we considered the interface agent, in tier two the sales agents are considered and in tier three all the inventory, factory and supplier agents are considered. Fig3 represents the architecture of the three tier setup. The whole system is implemented in Java 1.6 and is implemented on a LAN of thirteen machines, which comprises of Windows XP professional connected to 10 Mpbs LAN network.

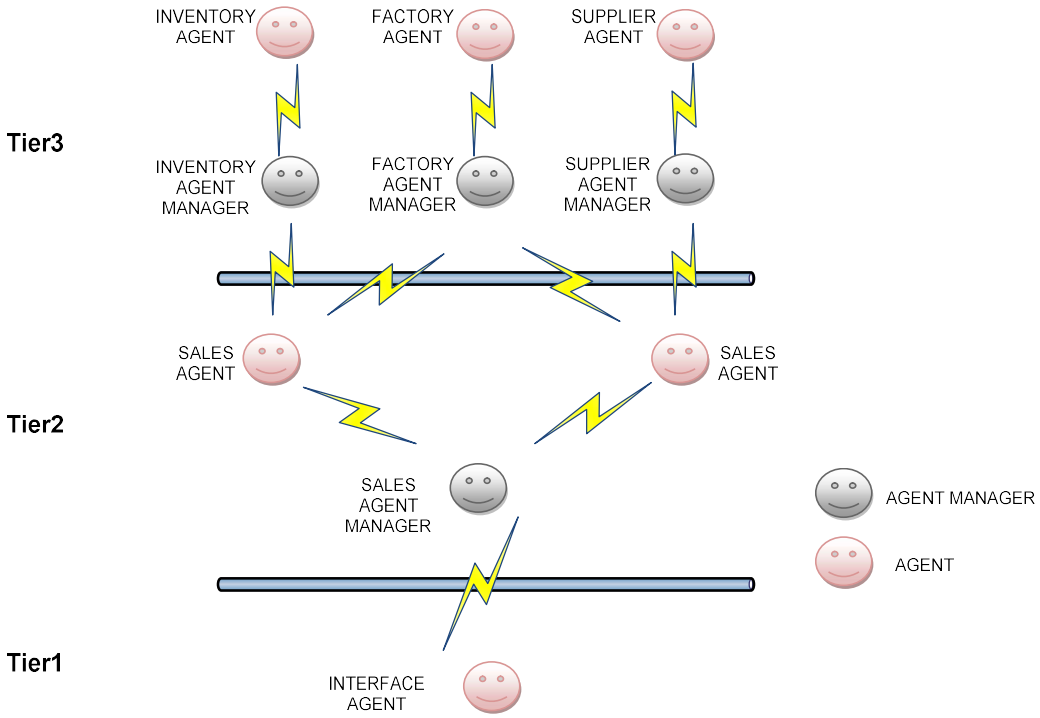

Fig.3.Representation of three tier architecture

4. Experimental Setup

The experiment was conducted in the Software Engineering Lab, Department of Computer Applications at M.S. Ramaiah Institute of Technology Bangalore. There is one server with fifteen terminals in the lab. The configuration of the server is, Model Name: IBM XSERIES 3650,2U rack server with quad core dual Xeon processor @ 2.0 GHZ, 8MB L2 Cache, 1333 MHz DUAL Gigabit Ethernet 10/100/1000 Embedded, 4GB DDR2 RAM, IBM SAS 8K RAID Controller with 256 MB CACHE IBM 10RPM, SAS 300GB Hard disk*3 No's =Total capacity 600GB. The configuration of the nodes used is , Model Name: IBM Think Center 8985 B63, Intel Core 2 DUO processor @ 2.2GHZ, 2 GB Ram, 160GB Hard disk, IBM Keyboard & Scroll Mouse, 20X DVD Writer & 19" IBM TFT Color Monitor L194 with the Internet Leased line with 60 Mbps data speed.

4.1 Execution Environment

The model is implemented using the software JADE, which is a Framework implemented in Java language which can be readily used for the implementation of multi-agent systems through a middleware that obeys with the Foundation for Intelligent Physical Agents (FIPA) specifications. It has a very good set of graphical tools that supports the debugging and deployment phases. The agent platform can be distributed across machines and the configuration can be controlled via a remote GUI. The configuration can be even changed at run-time by moving agents from one machine to another one, as and when required. JADE is completely implemented in Java language and the minimal system requirement is the version 1.4 of Java. Container is a running instance of the JADE running environment containing several agents. A single Main Container must always be active in a platform and all other containers register with it as soon as they start. Main Container holds two special agents Agent Management System (AMS) that provides the naming service; name, ensure uniqueness, create/destroy agents and Directory Facilitator (DF) that provides a Yellow Pages service by means of which an agent can find other agents providing the services he requires in order to achieve his goals.

4.2Results obtained from implementation

From the graphs, it is observed that in all the three different tiers considered, the interface agent has taken the highest response time, the supply agent has taken the least response time and the inventory agent has taken the average response time. The maximum response time taken by the single tier is 4000msec, the maximum response time taken by the two tiers is 3500 msec, and the maximum response time taken by the three tiers is 2000 msec. From this we can infer that we can achieve better response time by increasing the number

of tiers. Thus we can suggest that the experimental study helps to decide the number of tiers while deploying the agents in the system.

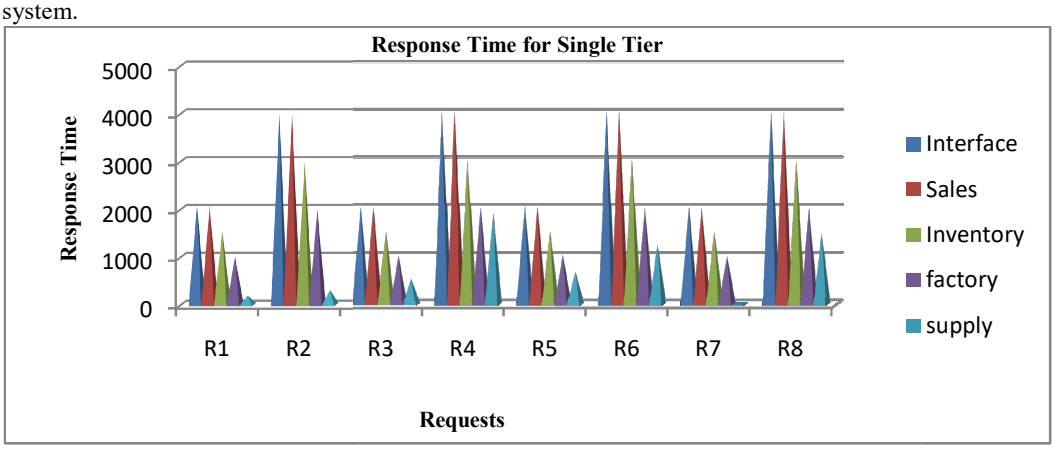

Fig..4.Response Time in Single Tier

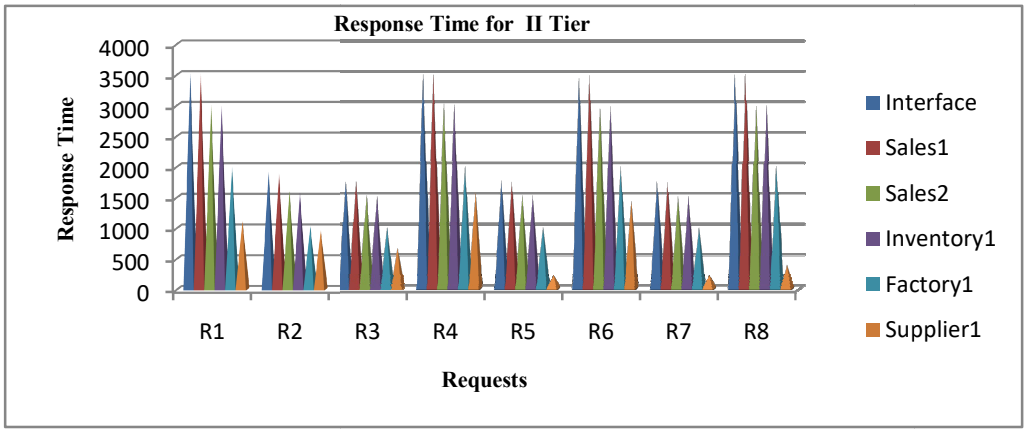

Fig.5. Response Time in Two Tiers

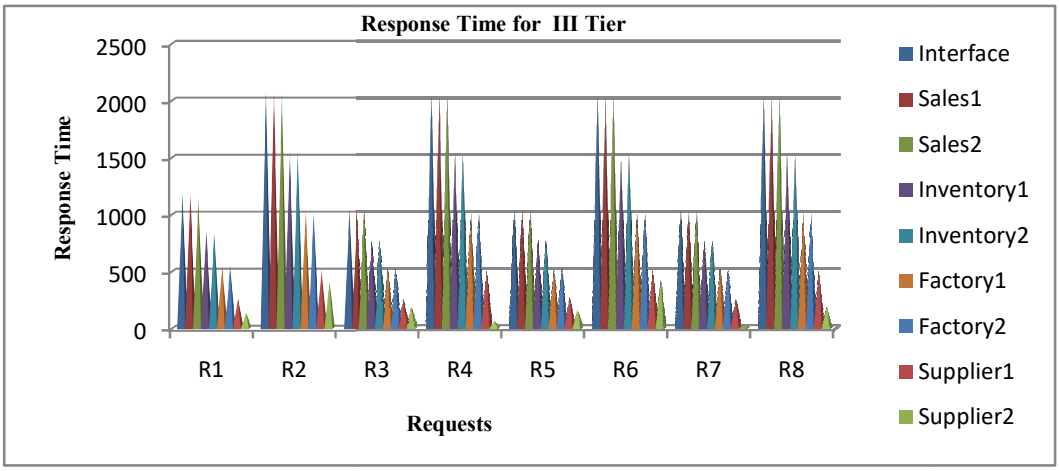

Fig.6. Response Time in Three Tiers

4.3 Comparison of Results with simulation results

We used the SMTQA[17] tool to validate the results obtained from the experimental study. We simulated for 1000 requests for the scenario one tier, two tier and three tier and the results are tabulated in Table1, Table2 and Table3 respectively. To compare with the experimental study, we considered the replicas in two tier and three tier. From the results, we observed that the response time for agents is better when we added replicas for the agents in both the cases. For the Interface agent we have not considered any replica and in all the three setup we got the same response time 0.010. In tier II we considered replicas for Inventory and Factory Agent and we got better

performance for these agents. In three tier, we considered two Sales Agent, three Supply Agent, two Inventory Agent and three Factory Agent and observed similar results. Thus, results show when more agents are used we get better performance.

Table1. Simulation Results for Tier One

Agents	Average Response Time
Interface Agent	0.010
Sales Agent	0.020
Supply Agent	0.023
Inventory Agent	0.025
Factory Agent	0.023

Table1. Simulation Results for Tier Two

Agents	**Average Response Time**
Interface Agent	0.010
Sales Agent	0.020
Supply Agent	0.0225
Inventory (2)Agent	0.012
Factory (2) Agent	0.011

Table2. Simulation Results for Tier Three

Agents	Average Response Time
Interface Agent	0.010
(2) Sales Agent	0.010
(3) Supply Agent	0.006
(2) Inventory Agent	0.008
(3) Factory Agent	0.006

5. CONCLUSION

We conducted this experimental study in our Software Engineering laboratory and the results are compared with the tool SMTQA. We implemented a case study using JADE environment. We considered a simple Supply-Chain Management system and executed the application by considering a one tier, two tier and three tier architecture. We implemented the application by deploying the agents in different physical machines. The obtained results are collected and the performance metrics response time is calculated for the different Agents. The results are validated by simulating the same scenario using the tool SMTQA and observed similar results for the Agents considered for discussion.

REFERENCES

[1] C.U. Smith and L.G. , Williams., Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software. s.l.: Addison Wesley, 2002.

[2] www.perfeng.com. Visited 17-4-2017[Online]

[3] Nicholas.R.Jennings, Katia Sycara, K.Wooldridge, M.: A roadmap of agent research and development. International Journal of Autonomous Agents and Multi-Agent Systems (1998) 7–38.

[4]PeterStone,Manuela eloso(2000),"Multi agent Systems: A Survey from a Machine Learning Perspective", In Autonomous Robotics volume 8,Number 3,July 2000. Carnegie Mellon University, Pittsburgh.

[5] T. Moyaux, B. Chaib-draa, and S. D'Amours, Multiagent based Supply Chain Management, Springer-Verlag, Berlin Heidelberg, 2006.

[6] Ramakrishna Govindu, Ratna Babu Chinnam (2007)," MASCF: A generic process-centered methodological framework for analysis and design of multi-agent supply chain systems", Computers & Industrial Engineering 53 (2007) 584–609.

[7] Chang-Hyun Jo , Jeffery M. Einhorn(2003),"A Process for BDI Agent-based Software Construction", IMCSE 2003 – SERP'03,Las Vegas, Nevada, USA.

[8] V. Cortellessa, G. Iazeolla and R. Mirandola,., "Early Generation of Performance Models for Object-Oriented Systems." 2000, Vol. vol.147, pp. pp. 61-72.

[9] S. Balsamo and M. Marzolla., Simulation-Based Performance Modeling of UML Software Architectures. Ph. D thesis, Ca' Foscari University of Venice. Italy : s.n., 2004.

[10] Simonetta Balsamo and Moreno Marzolla., "Performance Evaluation of UML Software architectures with Multiclass Queueing Network Models." Spain : s.n., July 2005. Proc. 5th International Workshop on Software and Performance . pp. pp. 37-42.

[11] Ka-Po Chow and Yu-Kwong Kwok "On Load Balancing for Distributed Multiagent Computing", ieee transactions on parallel and distributed systems, vol. 13, no.8, 2002. Object Management Group (OMG) UML 2.1.1 Superstructure Specification, 2007.

[12] Hyunsang Youn, Suhyeon Jang, Eunseok Lee, A Novel Approach for Performance Improvement of Multi-Agent based System Architecture, International Journal of Software Engineering and Its Application,Vol. 2, No. 1, January, 2008.

[13] Ye Chen , Yun Peng , Tim Finin , Yannis Labrou , Bill Chu , Jian Yao , Rongming Sun , BobWillhelm , Scott Cost, A negotiation-based Multi-agent System for Supply Chain Management ,In Proceedings of Agents 99 Workshop on Agent Based Decision-Support for Managing the Internet-Enabled Supply-Chain,pp 15-20.

[14] A. Ali, M. Aslam, J. I. Janjua and M.U. Chaudhry, Methodology for Performance Evaluation of Distributed Multi Agent System. The Nucleus 54, No. 2 (2017) 75-82.

[15] L. C. M. Perera & A. S. Karunananda. Using a Multi-Agent system for supply chain management,Int. J. of Design & Nature and Ecodynamics. Vol. 11, No. 2 (2016) 107–115.

[16] Kamalendu Pala, Bill Karakostas,A Multi Agent-Based Service Framework for Supply Chain Management ,Procedia Computer Science 32 (2014) 53 – 60.

[17]D E Geetha and T V S Kumar. Performance Modeling and Evaluation of Distributed Systems, Ph.D thesis, Visvesvaraiah Technological University, Karnataka, 2012