Int. J. of Advanced Networking and Applications
Volume: 02, Issue: 02, Pages:614-620 (2010)

Local Area Network Performance Using UML

Dr Vipin Saxena
Associate Professor, BabaSaseb Bhim Rao Ambedkar University (Central University), India
Email: vsax@rediffmail.com

Deepa Raj
Assistant professor, BabaSaseb Bhim Rao Ambedkar University (Central University), India
Email: deepa_raj200@yahoo.co.in

ABSTRACT

This paper explains the performance of LAN/WAN using UML. Nowadays, Local Area Network (LAN) System is an integral part of any organization. This may established by the use of the fiber optics cables or through Wi-Fi connectivity called as Wireless Local Area Network (WLAN). In the network connectivity, administrator has to ensure whether the destination node is receiving the data correctly, safely and within minimum time period. In the country, many of the organizations have the LAN or WLAN but one has to check the performance of these, therefore, in this paper, performance of LAN/WLAN is observed through the UML design. The node is considered as a stereotype and one process is transmitted from one machine to the other machine and data transmission rate is recorded through experimental study. A real case study of Babasaheb Bhimrao Ambedkar Central University is considered and results are depicted in the form of graphs and table

Keywords - LAN, Performance, UML, Wi-Fi , Process

Date of Submission: May 22, 2010 Date of Acceptance: August 6, 2010

1. Related Work

Less numbers of papers available in the field of UML design of architecture of object oriented Software System. Drozowski, M. [1] explained estimating execution time of distributed application in parallel processing and applied mathematic. DeMigu et al. [2] explained in his paper UML extension for the specification and evaluation of latency constraints in architectural Models. Holz [3] explained the application of UML within the scope of new telecommunication architectures. Hoeben [4] computed the performance of distributed system UML Models. Habib [5] described simulated analysis of server placement on network topology design. Helsinghar et al. [6] explained about tools and techniques used for performance measurement of large distributed multi agent system. Nitto [7] described how to derive executable process description from UML. Petriu and Jalnapurkar [8] described architecture-based performance analysis applied to a telecommunication system. Simonetta and Moreno [9] explained the performance evaluation of mobile systems through UML. Zhang et al. [10] estimated performance analysis of network topologies in agent based open connectivity architecture for decision support system. Yung Terng and Moris [12] explained load sharing for distributed system. Zhen Shen and Zhuang [11] explained how to improve network performance by traffic reduction.

2. Background

2.1 Distributed Computer System

A distributed system consists of multiple autonomous computers that communicate through a computer network. The computers interact with each other in order to solve the computational problem. A distributed program runs in a distributed system. In distributed computing, a problem is divided into many tasks, each of which is solved by one computer. In the distributed system, individual computers were physically distributed within some geographical area. Nowadays, it is used in a much wider sense, and physical computer interact with each other by message passing technique. A distributed system may have a common goal, such as solving a large computational problem. Alternatively, each computer may have its own user with individual needs, and the purpose of the distributed system is to coordinate the use of shared resources or provide communication services to the users.

The structure of the system i.e. network topology, network latency, number of computers is not known in advance, the system may consist of different kinds of computers and network links, and the system may change during the execution of a distributed program. Structure of the distributed computer system is shown in Fig. 1 given below where each node as an independent computer with own processor and memory.

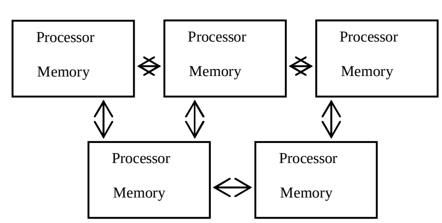

Figure 1 Structure of Distributed Computer System

Volume: 02, Issue: 02, Pages:614-620 (2010)

In distributed computing, each processor has its own private memory. Information is exchanged by passing messages between the processors. A distributed system is more reliable than a non-distributed system, as there is no single point of failure and it is very easier to expand and manage. Distributed algorithms can be analysed by communication operations. The simplest model of distributed computing is a synchronous system where all nodes operate in a lockstep fashion. During each communication round, all nodes in parallel receive the latest messages from their neighbors and perform arbitrary local computation, and then send new messages to their neighbors. In such systems, a central complexity measure is the number of synchronous communication rounds required to complete the task. This complexity measure is closely related to the diameter of the network.

2.1.1 Architectures of Distributed System

Various hardware and software architectures are used for distributed computing. At a lower level, it is necessary to interconnect multiple CPUs with some sort of network, regardless of whether that network is printed onto a circuit board or made up of loosely-coupled devices and cables. Distributed programming typically falls into one of several basic architectures or categories:

- **Client-server —** Client code contacts the server for data, then formats and displays it to the user. Input at the client is committed back to the server when it represents a permanent change.
- **3-tier architecture —** Three tier systems move the client intelligence to a middle tier so that stateless clients can be used. This simplifies application deployment. Most web applications are 3-Tier.
- **N-tier architecture —** N-Tier refers typically to web applications which further forward their requests to other enterprise services. This type of application is the one most responsible for the success of application servers.
- **Tightly coupled (clustered) —** Refers typically to a cluster of machines that closely work together, running a shared process in parallel. The task is subdivided in parts that are made individually by each one and then put back together to make the final result.
- **Peer-to-peer —** An architecture where there is no special machine or machines that provide a service or manage the network resources. Instead all responsibilities are uniformly divided among all machines, known as peers. Peers can serve both as clients and servers.
- **Space based —** Refers to an infrastructure that creates the illusion (virtualization) of one single address-space. Data are transparently replicated according to application needs. Decoupling in time, space and reference are achieved.

Another basic aspect of distributed computing architecture is the method of communicating and coordinating work among concurrent processes. Through various message passing protocols, processes may communicate directly with one another, typically in a master/slave relationship.

2.2 Peer to Peer Communication

Peer-to-Peer commonly abbreviated as **P2P** and it is method of communication in any distributed network architecture where each node composes of resources such as processing power, disk storage or network bandwidth which is directly available to other network participants, without the need for central coordination instances such as servers or stable hosts. Peers are both suppliers and consumers of resources, in contrast to the traditional client-server model where only servers supply and clients consume.

2.2.1 Architecture of P2P Systems

Peer-to-Peer networks are typically formed dynamically by *ad-hoc* additions of nodes. In an 'ad-hoc' network, the removal of nodes has no significant impact on the network. The distributed architecture of an application in a peer-to-peer system provides enhanced scalability and service robustness. Peer-to-peer systems often implement an Application Layer overlay network on top of the native or physical network topology. Such overlays are used for indexing and peer discovery. Content is typically exchanged directly over the underlying Internet Protocol (IP) network P2P networks are typically used for connecting nodes via largely ad hoc connections. Sharing content files containing audio, video, data or anything in digital format is very common, and real time data, such as telephony traffic, is also passed using P2P technology.

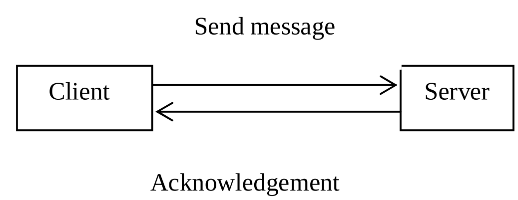

Figure 2 P2p Communication Technology

A pure P2P network does not have the notion of client or servers but it contains number of peer nodes that simultaneously work as both "clients" and "servers" to the other nodes on the network environment. The P2P overlay network consists of all the participating peers as network nodes. There are links between any two nodes that know each other i.e. if a participating peer knows the location of another peer in the P2P network, then there is a directed edge from the former node to the latter in the overlay network. Based on how the nodes in the overlay network are linked to each other, we can classify the P2P networks as unstructured or structured.

3. UML Model for Process Request

For designing the UML model and for finding the performance of network computer system in the distributed environment, process is running concurrently, the definition of process has defined in the fig 3. Each node is considered as a process which has a unique identity and it executed concurrently in the distributed environment. The communication among the processes in the distributed environment can be handled by the synchronization technique if the common resources are available at the client computer system. There are two type of communication – P2P communication which have been described in the background section and the another technique is broadcast protocol method. UML class diagram of P2P and broadcast signal are given in Fig 4 and 5.

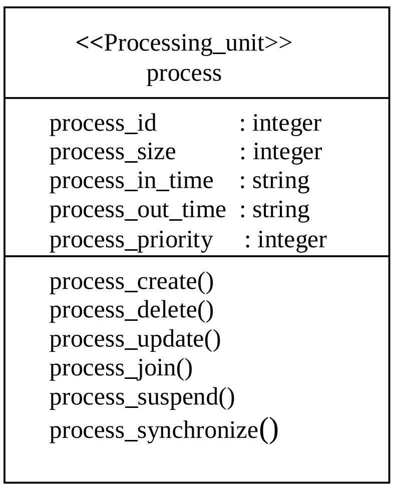

Figure 3 Class Diagram of Process

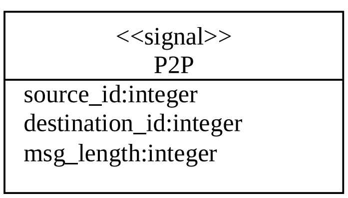

Figure 4 UML Class Diagram of P2p Communication

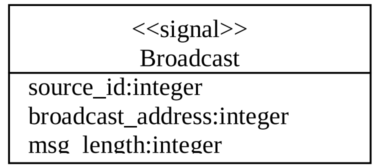

Figure 5 UML Class Diagram of Broadcast

UML class diagram of process request to the server is depicted in Fig. 6. Major classes involved in distributed network are shown in this figure. The PEC (Process Execution Controller) is responsible for process execution with the help of processor and memory. PEC will interact with signal class for sending request message to the server which may be type P2P and broadcast signal.

The UML class diagrams for both the signal have been previously explained in this section. Signal class will also interact with Communication_Channel class for sending message in the form of signals. Through Communication_Channel these signals are sending to a switch. The Len_Plex_Switch class is responsible for deciding the path to the DNS for resolving the host. After resolving the IP address of the server the information will be submitted to the Load_distributor class. The Load_distributor class is entirely responsible for allocating appropriate server and completing the process request. The server allocation is based on various factors like graphical location, network load and server availability.

The UML sequence diagram of process request to the server is shown in the Fig 7. This diagram shows the message passing technique between the classes as well as life line of each object during execution.

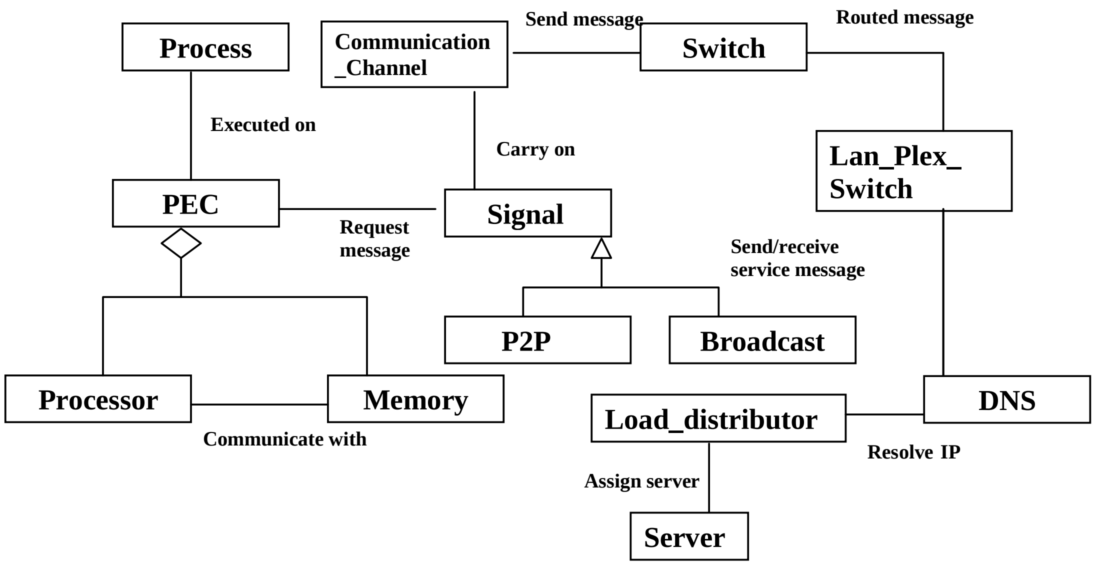

Figure 6 UMLClass Diagram of Process Request

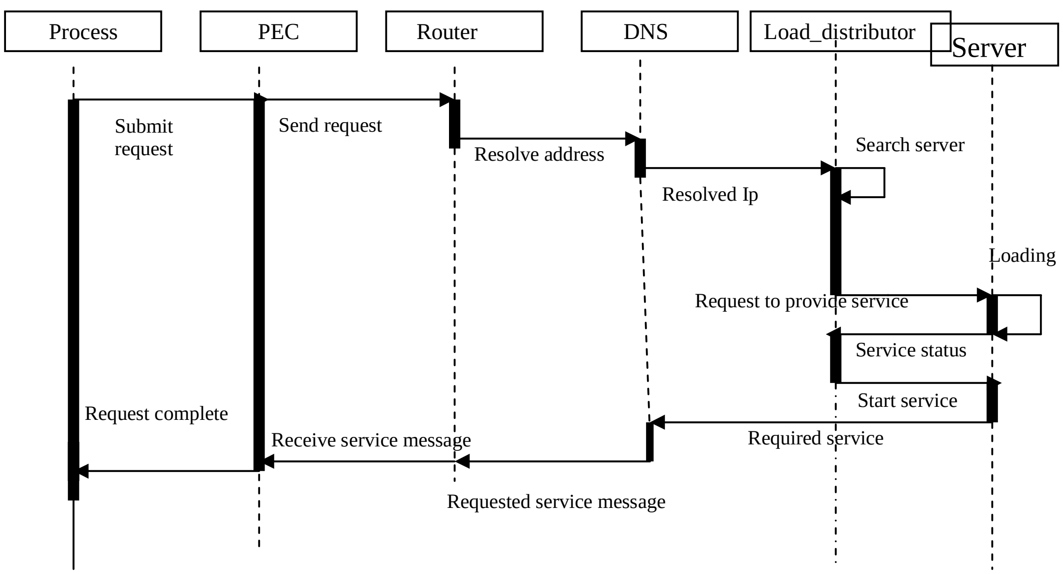

Figure 7 UML Sequence Diagram of Process Request

4. Network Performance Evaluation: An Experimental Case Study

For measuring the performance of the network we considered Local Area Network (LAN) of BabaSaheb BhimRao Ambedkar Central University India as a case study.

The lay out of network system is displayed in the Fig 8, where each node has a processor of type Core 2 Duo.

For measuring the performance of network structure we considered Bytemon software which is a network and resource performance monitoring application, intended to give administrators real-time information about the condition of network and servers. It can be used to monitor resource availability and usage as well as performance of data of all kinds, such as network performance parameters (bandwidth usage, protocols

usage), the availability of network resources, and the network latency of access to remote devices and services

Bytemon software has following uses:

- **Traffic Monitoring**: It monitors the bandwidth and usage monitoring networking equipment such as hubs, routers and switches
- **Network Protocol Usage Analysis**: It is also used to analyze network traffic by protocols, source/destination IP etc.
- **Performance Monitoring**: It monitors the performance of server and any networked devices network printers, network storage
- **Availability and Latency Monitoring**: It also monitors remote devices and servers like mail servers or web servers.

In this Network, all switches follow packet-switching technique.

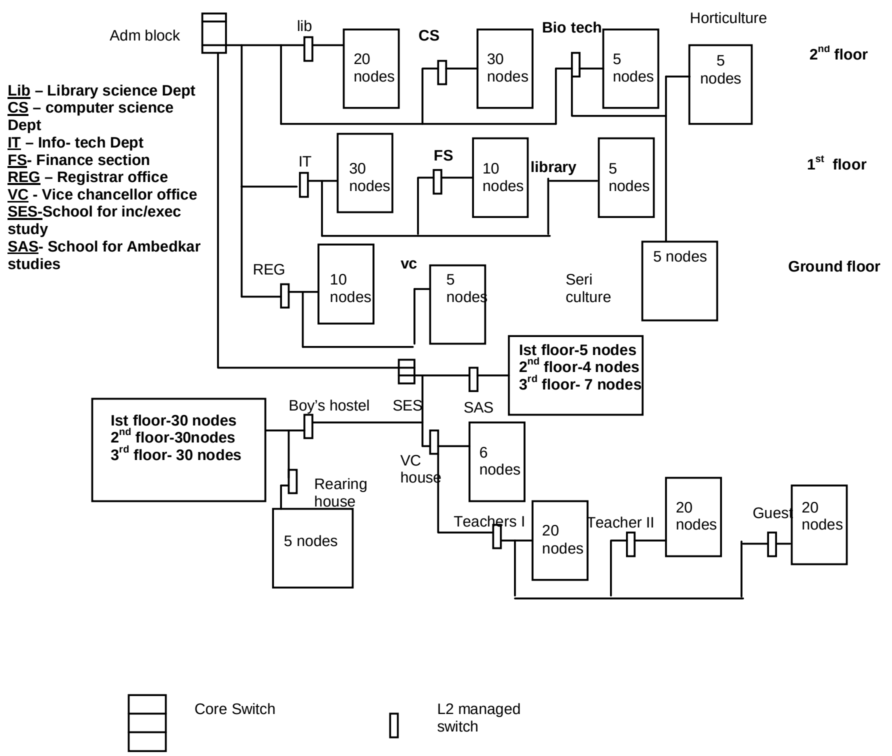

Figure 8 Layout of Local Area Network

Let us perform some test to evaluate the performance of distributed network by generating a packet from one node to another node. Table 1 and Table 2 display information about the network traffic out i.e. how much packets are send from the server for one minute time interval for the two sets of readings. With the help of these readings we can find out the transmission rate of the traffic out.

Table 1 Network Traffic Out (First Set)

S.N0	First Minute	Second Minute	Third Minute	Fourth Minute
1	741.5	146.5	48	33
2	256	876.5	37	37
3	219.5	37	33	411
4	252	33	102	31
5	252.5	64	91	37
6	256	31	31	37
7	250.5	77.5	31	37
8	312.5	159.39	37	152.5
9	109.5	1150	69.5	514.5
10	371	27	74.5	293.5
Average	**302.10**	**260.19**	**55.40**	**158.35**

Table 2 Network Traffic Out (Second Set)

S.N0	First minute	Second Minute	Third Minute	Fourth Minute
1	38.5	329.5	136	1412
2	167	927	66	3217
3	58	37	167.5	2146
4	172	166	108	1071
5	33	212	100	2215
6	53	200	341	70
7	212	341	27	37
8	46	33	78.5	37
9	327.5	444	186.5	189.5
10	108	37	60.5	89.5
Average	**121.50**	**272.65**	**127.10**	**1048.40**

Taking average of data of these two tables for each sets of time, data are displayed in Table 3

Table 3 Network Traffic Out (Average)

S.NO	First Minute	Second minute	Third Minute	Fourth Minute
1	121.50	272.65	127.10	1048.40
2	302.10	260.19	55.40	158.35
Average	**211.80**	**266.42**	**91.25**	**603.38**

Fig. 9 displays how much average packets coming out from server in the network graphically per unit time in minute.

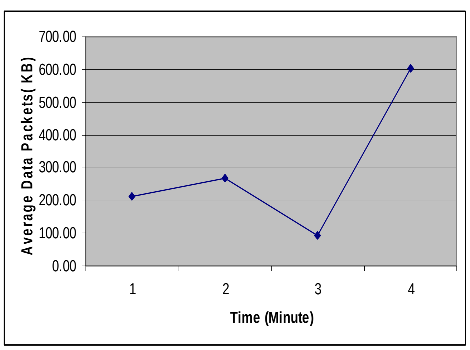

Figure 9 Network Traffic Out

Table 4-6 display information about how much packets inserted or how many request send to the server through the network per unit time.

Table 4 Network Traffic In (First Set)

S.NO	First Minute	Second Minute	Third Minute	Fourth Minute
1	0.5	0.5	0.5	0.5
2	12.5	3	0.5	0.5
3	20	0.5	0.5	5
4	6	2	6	0.5
5	0.5	0.5	0.5	0.5
6	0.5	0.5	0.5	1.5
7	0.5	0.5	0.5	0.5
8	0.5	11	0.5	0.5
9	0.5	1	0.5	0.5
10	0.5	1	0.5	1
Average	**4.20**	**2.05**	**1.05**	**1.10**

Table 5 Network Traffic In (Second Set)

S.NO	First Minute	Second Minute	Third Minute	Fourth Minute
1	0.5	0.5	0.5	0.5
2	12.5	3	0.5	0.5
3	20	0.5	0.5	5
4	6	2	6	0.5
5	0.5	0.5	0.5	0.5
6	0.5	0.5	0.5	1.5
7	0.5	0.5	0.5	0.5
8	0.5	11	0.5	0.5
9	0.5	1	0.5	0.5
10	0.5	1	0.5	1
Average	**4.20**	**2.05**	**1.05**	**1.10**

Table 6 Network Traffic In (Average)

S.NO	First Minute	Second Minute	Third Minute	Fourth Minute
1	4.20	2.05	1.05	1.10
2	4.20	2.05	1.05	1.10
Average	**4.20**	**2.05**	**1.05**	**1.10**

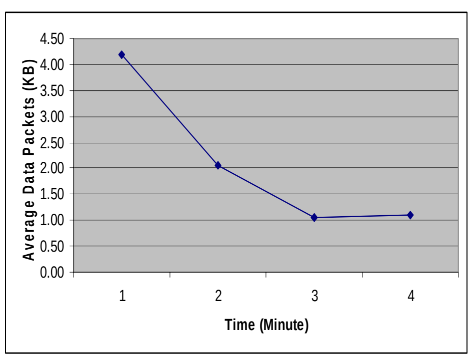

Figure 10 Network Traffic In

Latency is the time delay caused by getting a network message to a remote computer, and getting a response back again. It is normally expressed in milliseconds (thousandths of a second). Table 7 displays pinging latency of the network. It displays time delay in millisecond. Fig. 11 displays pinging latency in the form of Graph.

Table 7 Pinging Latency

S.NO	First Minute	Second Minute	Third minute	Fourth Minute
1	0.8	2.8	0.7	1.6
2	0.2	0.6	0.3	2.8
Ave	**0.50**	**1.70**	**0.50**	**2.20**

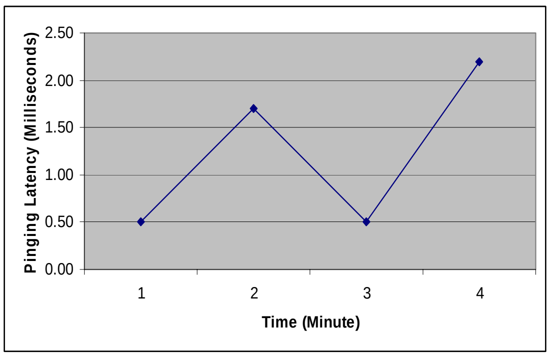

Figure 11 Pinging Latency

5. Conclusions

By applying the performance measuring tool we find that in the Babasaheb Bhimrao Ambedkar Central University, (Network traffic in) is less than (Network traffic out). It means that server perform more download than upload. By measuring pinging latency it was found that time delay is maximized up to 2.20 ms.

Acknowledgements
Authors are very thankful to Prof. B. Hanumaiah,Vice Chancellor, Baba Saheb Bhimrao Ambedkar University(A Central University), Vidya Vihar, Rai Bareli Road, Lucknow, India, for providing the excellent computation facilities in the University Campus. Thanks also due to the University Grant Commission, India for providing financial assistance to the University for the Research.

References

[1] Drozowski, M. (2002), “Estimating Execution Time of Distributed Application Parallel Processing and Applied Mathematics”, 4th International Conference PPAM 2001, LNCS 2328, Springer – Velag, p.p. 137-142.

[2] DeMiguel, M., Lambolais, T., Hannouz, M. and B-brezetz, S. (2000), “ UML Extension for the Specification and Evaluation of Latency Constraints in architectural Models “, in proceedings of WOSP 2000, p.p. 83-88.

[3] Holz, E. (1997), “Application of UML within the Scope of New Telecommunication Architectures”, GROOM Workshop on UML, Mannheim: Physicaverlag.

[4] Hoeben, F. (2000), “Using UML Models for Performance Calculation”, In Proceeding of WOSP 2000, p.p. 77-82.

[5] Habib, S. J (2005), “Simulated Analysis of Server Placement on Network Topology Design”, Proceedings of the 3rd ACS/IEEE International Conference on Computer System and Application, Cairo Egypt, pp. 80-87.

[6] Helsinger, A., Lazarus, R., W. Wright, W. and J. Zinnky, J. (2003), “Tools and Techniques for Performance Measurement of Large Distributed Multi Agent System”, Proceedings of AAMAS 03 Conference, Australia, pp. 843-850

[7] Nitto, E. (2002), “Deriving Executable Process Description from UML”, ICSE 19-25, Orlando, Florida, USA.

[8] Petriu, D. S. C, and Jalnapurkar, A. (2000), “Architecture-Based Performance Analysis Applied to a Telecommunication System”, *IEEE Transactions on Software Engineering*, Vol. 26, No. 11. P. 1049-1065.

[9] Simonetta, B. and. Moreno,M. (2003), “Towards Performance Evaluation of Mobile Systems in UML”, In B. Di Martino, L. T. Yang, and C. Bobenau, editors, The European Simulation and Modeling Conference, Retrieved from www.dsi.unive.it/~marzolla/papers/marzolla-esmc03-final.ps.gz.

[10] Zhang, H. L., Leung, H. C. and Raikundalia, G. K (2008), “Performance Analysis of Network Topologies in Agent Based Open Connectivity Architecture for DSS”, In Proceedings of the 27 nth International Conference on Advance Information Networking and Application.

[11] Zhen Shen Guo and Zhuang, Y. (1997), “Improving Network Performance by Traffic Reduction”, In Proceeding of Information Communication and Signal Processing, ICICS, Vol. 2, PP. 1226-1230.

[12] Yung-Terng Wangand and Moris, R.T.J (1985) “Load Sharing in Distributed System”, *IEEE Transaction on Computers*, Vol. C-34, Issue 3, pp. 204-217.