DESIGN AND IMPLEMENTATION OF AMODIFIED PULL-ALL MIGRATION STRATEGY IN AGLETS MOBILE AGENT FOR EFFECTIVE NETWORK-LOAD MANAGEMENT

[1]Osunade O.,[2]Oyewole, S. A, [3]Azeez N. A, [4]Osofisan A, [5]Akinola M.I

[1,4,5]Department of Computer Science, University of Ibadan, Ibadan, Nigeria.
[2]Department of Information Technology,
Durban University of Technology (DUT), Durban, South Africa.
[3]Department of Computer Sciences, University of Lagos, Lagos Nigeria.

ABSTRACT
The conventional Simple Network Management Protocol (SNMP) approach used in client and server architectu which runs in a centralized manner has one common challenge, as network size increase, the number of potenti problems and complexity of the network task also increase. Although the centralized management approach giv network administrators a flexibility of managing the whole network from a single place, it is prone to informatic overloading, excessive processing load and heavy usage of network bandwidth. In this situation, effective netwo management is required. Contrary to client-server, Mobile Agents handles such situation by shipping necessary cod and its execution state closer to the data which often reduces network load and generates improve system performanc However, despite all these useful advantages, past literatures have shown that Mobile agents still have some inhere drawbacks, because of its poorly coupled migration model and the limitations of static and mixture approa commonly used to reduce network load. Therefore, effective implementation of a data migration strategy in a giv distributed network is critical in reducing network load and the holistic performance of the computer Network. T purpose of this study is therefore to design an approach based on network deployment of the modified pull-all co data migration pattern using Java based Aglets transfer protocol (ATP) that send and retrieve information. T transmission time and network load at each migration nodes are noted.Our results show that the proposed approa can be efficiently used for information retrieval in a distributed environment.

Keyword: Aglet Mobile agent, Network-load, SNMP, Client and Server, Pull-All

INTRODUCTION

As the network size increases, the number of potential problems and complexity of the network generates significant traffic that overloads Network management station (Kona & Xu, 2002). Mobile agent frameworks have attracted a lot ofattention in recent years; most especially on how the network efficiency can be improved upon. Many researchers see mobile agent as an option in this situation to distribute and efficiently manage network traffic (Adhicandra et al. 2003;Zhen et al , 2006 ;ZhiLong, et al (2012) ; Osunadeet al 2013, Madkour, et al. 2014).Contrary to other mobile code paradigms such as client-server, remote evaluationand code on demand paradigms(Carzaniga et al. 1997), Mobile Agent handles such situation by shipping necessary code and its execution state, closer to the data, which have been found to often reduce network load and generates improve system performance(Braun et al. (2001, 2005), Hunt, 2002, Bernich, &Mourlin, 2006). This process is referred to as migration strategy (Powell & Miller, 1993). Code migration strategy can occur in two states; either part/unit of the code or the whole is push (sent) or pull (download) over the network (Braun et al. 2001).

Each of these above strategies can be implemented using either asynchronous or synchronous protocols. In Aglets migration strategy, just like any other Java-based mobile agent, accumulated data collected from its previously visited nodes (Xuhui, et al 2006) are pushed and added to its existing data until it reaches its destination node. Fundamentally, this has huge potential to increase network latency time and delay in delivery time as the load size increases. This process is also subjected to other problems such as loss of data(Osunade&Atanda 2008), untimely termination of mobile agent before reaching its destination.

The current evolving network and compu technology, coupled with the exponential growth of services and information available on the inter demand migration process that can reduce netw traffic, traffic congestion, low transmission time effective bandwidth utilization. This system performance can be improved (Kim & Feamster, 2013) by carrying out a rework on the network load and transmission time process as well as feature of the chosen software agents, such as Aglets (Braun et al, 2001). The process of moving from one system in the network manager involves wastage of time and with the use of mobile agents, there would be no time loss also, there would be increased productivity and efficiency of the network.

This paper focuses on the pull code data migration pattern of network systems, with the main focus on how to design a model for the strategy and implement it in an existing Aglets mobile agent. Apart from the fact that java is platform dependent, the language also has direct supports for sockets, RMI, threads and object serialization (Buchanan, et al 1999). The purpose of this study is to solve the problem that deal with the migration of network from one computer system to another computer system in the Aglet network in order to gather information. The transmission time and network load will be used as the experimental metrics.

The rest of this paper is organised in the following way. In section 2, extensive literature review on mobile agent implementation and performance issues in network management is presented. In section 3, detailed explanation of the code migration strategy in Aglet is explained. In section 4, the authors describe the proposed modified pull-all migration strategy architecture in Aglet mobile agent for network management. Section 5 presents an aglet base java implementation of the migration processes. We conclude the paper in section 6 and remarks on future work is in section 7.

Literature Review

Mobile agents migrate with their codes, data and state information. When mobile agents execute tasks on hosts they also generate data that are migrated along with the mobile agent. The data size migrated by the mobile agent affects the performance of the mobile agent, thus how it is migrated will determine the completion time of the task assigned to the mobile agent (Osunade, 2007). However, there are cases where mobile agents perform less than client-server techniques, example of such task that require high bandwidth is information filtering.

Accounts from literature has it that static and mixture approaches are the two major ways that can be used to reduce network load(Carzaniga, et al. (1997); Vigna(1998); Baldi et al (1997); Bandi&Picco(1998); Braun, (2003)). However, due to the fact that static approach are not generally sufficient for all kinds of applications and the mixture approach is undependable because it heavily relies on the assumption of the values derived from the network bandwidth, latency request and result size. In the past few years, many researchers have investigated performance of mobile agent as it relate to network load and timing.

Gupta &Kansal (2011), describe theoretical views on application features of six mobile agents. Oyewole et al, 2012 presented qualitative comparison among three mobile agents (Aglet and Tracy, Jade) and quantitative analysis on Aglet and JADE using transmission time and compression time as performance metric. Gavalas, et al. 1999 presented the application of mobile agent in bulk transfers of data, using bandwidth utilisation as the metric. Carzaniga et al. (1997) and Vigna (1998) developed a simple performance model for the information retrieval domain.

In Osunade et al 2013 a prototype of SIMMASS which demonstrate the basic feature of a mobile agent that has capability to provide programmers with a Library for writing mobile agent applications and an environment in which to execute the agents was presented. In Iqbal et al. (1998) several algorithms to compute the optimal migration sequence of a single agent were presented. Kona & Xu, 2002 presented a hybrid model based on mobile agent and SNMP strategies for efficient management of heterogeneous networks. The framework gives management stations or network administratorthe flexibility of using any approach according to the characteristics of target network. In Lefebvre et al (2006) the researchers proposed a network management framework using mobile agents' as a diagnostic tools that are able to find more precisely a cause of a network failure by finding alternate paths to gather more data about the failure. Osunade, 2007, developsa mathematical model for the performance evaluation of data migration pattern in mobile agent systems. The two (2) stages modified data migration strategies uses transmission time and network load as metric parameters.

This paper is distinctively different from all the above, as it dwell much on how to model and implement pull-all migration strategy in Aglet mobile agent in order to reduce Network-load, using Java programming language as an implementation tool to demonstrate the efficiency of the model.

Code Migration Strategy in Mobile Agents

A mobile agent system provides the execution environment for mobile agents. Some of the common mobile agent Systems that have been developed in the past are: Telescript, Concordial, Ara, Jade, Semoa, Tacoma, Tracy, Ajanta, Aglets and so on. Aglet is one of the first complete Internet agent systems to be developed on the Java class library. Aglet was developed by the IBM Tokyo Research Laboratory. An aglet requires a host Java application, an "aglet host" to be running on a computer before it can visit that computer. An aglet can experience many events in its life cycle. It can be created, cloned, dispatched, retracted, deactivated, activated and disposed. The Java aglet extends the model of network-mobile code made famous by Java applets. Like an applet, the class files for an aglet can migrate across a network. But unlike applets, when an aglet migrates it also carries its state. An aglet is a running Java program (code and state) that can move from one host to another on a network. In addition, because an aglet carries its state wherever it goes, it can travel sequentially to many destinations on a network, including returning back to its original host (Venners, 1997).

Code migration is the movement of programming code from one system to another and the method used to transfer mobile code over the network is called migration strategy. During the transfer it is not only the code that has to be shipped but also the execution state of the mobile agent and data the agent carry along.

There are various migration strategies. One of the strategies is to send the complete program (whole code) over the network. Another strategy is to transmit only certain required parts (units) of the code. Another choice is either to push code over the network, that is, when code is sent over the network in advance, or to pull (download) code from a reachable location, while the mobile agent (execution unit) loads code from some suitable source. If the push code variant is used, code could be sent to the next location only, or to all locations on the agent's itinerary.

Braun et al (2003) identifies two classes of migration strategies called pull and push strategies. The first strategy, called pull, uses a built-in Java technique to download classes on demand. The second class of migration strategies is named push strategies, where the mobile agent code and the serialized agent state are transmitted at once.

According to Erfurth et al, 2001, Migration strategy can be classified by answering the following finding questions:

- How much code is transmitted?
- When the code is sent (transmitted)?
- Where is the code transmitted to?

In spite of the selected strategy, the execution state of the mobile agent, code and the data must be transmitted.Formulas and other relevant information on above mentioned mobility models are detailed in Osunade & Osofisan 2008 and Braun et al, 2003.

Proposed Modified Pull-all Code Migration Strategy in Aglet.

The aglet object model was designed to benefit from the agent characteristics of Java while overcoming some of the deficiencies in the language system. In the model, a mobile agent is a mobile object that has its own thread of control which is event-driven, and communicates by message passing (Lange & Oshima, 1998). The strategy (S), which is pull-all-code, determines how the migration process occurs.The network consists of an ordered set of hosts, represented as:

L = L1 ….Lm.

The migration process is divided into three basic stages representing distinct stages of the of the migration process that can be modelled

- ***lve***: The migration of the mobile agent from the home host (mobile agent system) to the first host on the mobile agent's itinerary i.e. the agent leaves (*lve*);
- ***mig:*** The migration between hosts on the itinerary following the specified order μσ if from the home host i.e. the agent migrates (*mig*);
- ***home:*** The migration of the mobile agent back to its home host (mobile agent system) after completion of the itinerary i.e. the agent returns home (*home*);.

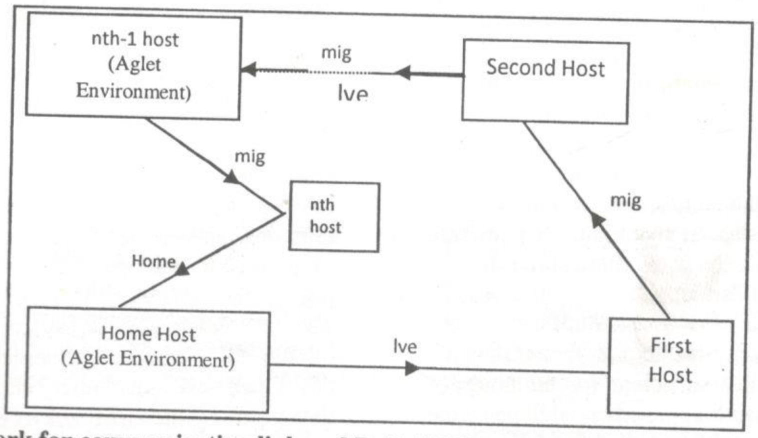


Fig. 1: Framework for communication link and Pull all Migration Strategy in Aglet Mobile Agent

This model uses the network load (B) and transmission time (T) as the metrics for performance evaluation to an ordered set of hosts, represented as $L = L_1, \ldots\ldots L_m$. The agent consists of some class files which can be dynamically loaded during execution from the agent's home host.

To implement the network load (B), the following notations and assumptions were made

- an agent consists of *n* units of code or *n* class files,
- each unit of code has a length of size B_c^k, k=1,…n,
- the sum of all code units is B_c
- an agent has a data of length of size B_d,
- an agent's state information is of length or size B_s
- a request to load a specific code unit has a length or size B_r
- The probability of dynamically loading code unit k on a host is P^k
- On a host the agent's data increases by d_L bytes

The migration is from La to La+1 where a =1,….,m-1. This is a host on the itinerary, but not the home host.

The pull-all code migration strategy (PullAll) is characterized by the request for code by the mobile agent on each host it visits. The mobile agent requests for all the code units whether or not they are all required on that host.

When the mobile agent leaves the home host for the first host on the itinerary, the network load is given as:

$$B_{ivs}(L,S) = B_c + B_r + B_s + B_d$$

As the mobile agent moves from one host to the next host on the itinerary, it migrates with the results of its activities on each host. The network load at this stage is thus:

$$B_{mig}(L,a,S) = B_d \sum_{i=1..a} dLi(1-\partial) + B_s + B_r + B_c$$

The network load of the mobile agent when it is returning to the home host is given by the equation

$$B_{home}(L,S) = B_d \sum_{i=1..a} dLi(1-\partial) + B_s$$

Transmission time when the mobile agent leaves the home host for the first host on itinerary is given as:

$$T_{ivo}(L,S) = 2\mu(B_d + B_s) + \partial(L_h, L_1) + \frac{(B_d + B_s)}{\tau(L_h, L_1)} + \frac{(B_r + B_c)}{\tau(L_1, L_h)}$$

When the mobile agent migrates from host to host on the itinerary, it migrates with the accumulation of results generated by the activity of the mobile agent. Thus, the transmission time is represented as:

$$T_{mig}(L,a,S) = 2\mu\left(B_d + \sum_{i=1..a} dLi(1-\partial) + B_s\right) + \partial(L_a, L_{a+1}) + \frac{(B_d + B_s) + \sum_{i=1..a} dLi}{\tau(L_a, L_{a+1})} + \partial(L_a, L_{a+1}) + \frac{(B_r + B_c)}{\tau(L_{a+1}, L_h)}$$

The time taken to return to the home host from the last host visited on the itinerary is given by the equation:

$$T_{home}(L,S) = 2\mu(B_{home}(L,S) + \partial(L_m + L_h) + \frac{(B_{home}(L,S)}{\tau(L_m, L_h)}$$

Implementation

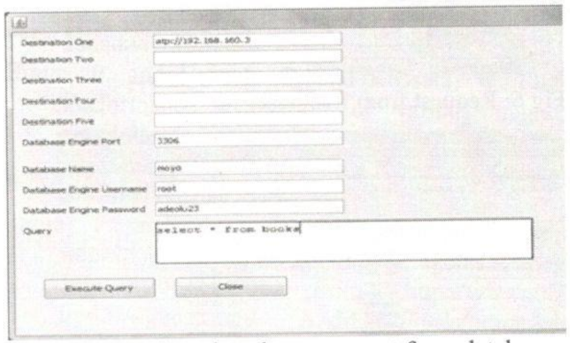
Requirements

In addition to the mathematical formulation for Pull-All modified migration strategy, the implementation of the model was carried out with JAVA programming language using the following software descriptions:

- Text Editor: The text editor used to input the source code for the aglets classes was EditPlus Text Editor V2.11 (1052).
- Java Development Kit (JDK 7U45- windows-i586)
- My SQL server (version 5.1.61- win 32) and SQLyog Community (version 11.2.0-3x86): they are used to create database and tables.
- IBM compatible microcomputer system with Ram of 2GB and processor speed 1.6GHZ.
- Aglets 2.0.1: This is an Aglets jar file that uses Tahiti server environment as its platform for execution.

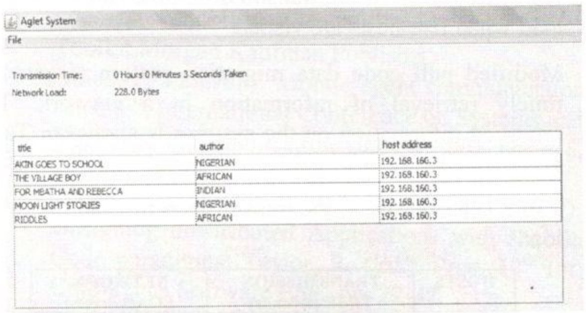
Results

The interface shown in figure below is the window displaying the interface of the request from host "atp://192.168.160.3".



Destination One	atp://192.168.160.3
Destination Two	
Destination Three	
Destination Four	
Destination Five	
Database Engine Port	3306
Database Name	moyo
Database Engine Username	root
Database Engine Password	adeolu23
Query	select * from books

Figure 2: Interface that shows request from database

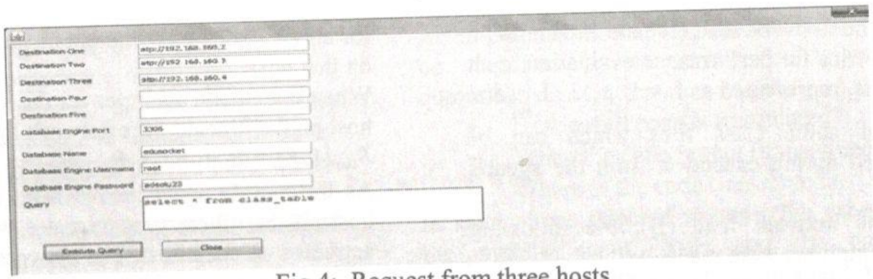


Transmission Time: 0 Hours 0 Minutes 3 Seconds Taken

Network Load: 228.0 Bytes

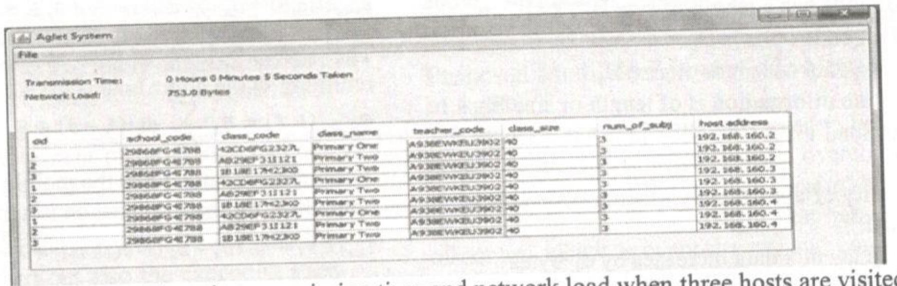
title	author	host address
AKIN GOES TO SCHOOL	NIGERIAN	192.168.160.3
THE VILLAGE BOY	AFRICAN	192.168.160.3
FOR MEATHA AND REBECCA	INDIAN	192.168.160.3
MOON LIGHT STORIES	NIGERIAN	192.168.160.3
RIDDLES	AFRICAN	192.168.160.3

Fig 3: Retrieved result, transmission time and network load when one host is visited.



Destination One	atp://192.168.160.2
Destination Two	atp://192.168.160.3
Destination Three	atp://192.168.160.4
Destination Four	
Destination Five	
Database Engine Port	3306
Database Name	edusocket
Database Engine Username	root
Database Engine Password	adeolu23
Query	select * from class_table

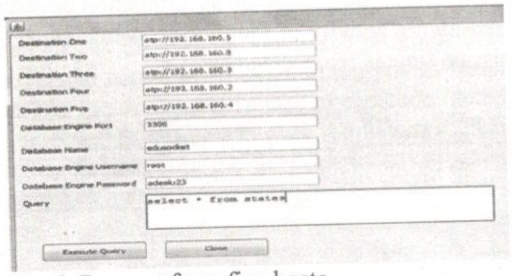
Fig 4: Request from three hosts



Transmission Time: 0 Hours 0 Minutes 5 Seconds Taken
Network Load: 753.0 Bytes

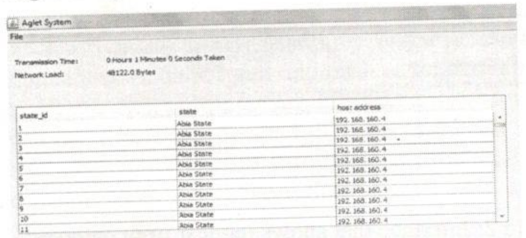
cid	school_code	class_code	class_name	teacher_code	class_size	num_of_subj	host address
1	29868FG4E788	42CD6FG2327L	Primary One	A938EWKEU3902	40	3	192.168.160.2
2	29868FG4E788	AB29EF311121	Primary Two	A938EWKEU3902	40	3	192.168.160.2
3	29868FG4E788	1B1BE17HG2300	Primary Two	A938EWKEU3902	40	3	192.168.160.2
1	29868FG4E788	42CD6FG2327L	Primary One	A938EWKEU3902	40	3	192.168.160.3
2	29868FG4E788	AB29EF311121	Primary Two	A938EWKEU3902	40	3	192.168.160.3
3	29868FG4E788	1B1BE17HG2300	Primary Two	A938EWKEU3902	40	3	192.168.160.3
1	29868FG4E788	42CD6FG2327L	Primary One	A938EWKEU3902	40	3	192.168.160.4
2	29868FG4E788	AB29EF311121	Primary Two	A938EWKEU3902	40	3	192.168.160.4
3	29868FG4E788	1B1BE17HG2300	Primary Two	A938EWKEU3902	40	3	192.168.160.4

Fig 5: Retrieved result, transmission time and network load when three hosts are visited



Destination One	atp://192.168.160.5
Destination Two	atp://192.168.160.8
Destination Three	atp://192.168.160.3
Destination Four	atp://192.168.160.2
Destination Five	atp://192.168.160.4
Database Engine Port	3306
Database Name	edusocket
Database Engine Username	root
Database Engine Password	adeolu23
Query	select * from states

Fig 6: Request from five hosts



Transmission Time: 0 Hours 1 Minutes 0 Seconds Taken
Network Load: 48122.0 Bytes

state_id	state	host address
1	Abia State	192.168.160.4
2	Abia State	192.168.160.4
3	Abia State	192.168.160.4
4	Abia State	192.168.160.4
5	Abia State	192.168.160.4
6	Abia State	192.168.160.4
7	Abia State	192.168.160.4
8	Abia State	192.168.160.4
9	Abia State	192.168.160.4
10	Abia State	192.168.160.4
11	Abia State	192.168.160.4

Fig 7: Retrieved result, transmission time and network load when five hosts are visited

Modified pull code data migration pattern targets the timely retrieval of information in a network. The retrieved information on the systems is shown in Table 1.

Table 1: Table shows the transmission time and network load

HOSTS	TRANSMISSION TIME (seconds)	NETWORK LOAD (bytes)
One	3	228
Two	3	531
Three	5	753
Four	8	975
Five	60	48122

Table 2: Comparison of network load

Σ	Pull All (PA)	Modified pull all (M P A)
0	578	463
0.1	562.5	460.5
0.2	547	458
0.3	531.5	455.5
0.4	516	453
0.5	500.5	450.5
0.6	485	448
0.7	469.5	445.5
0.8	454	438.5
0.9	438.5	440.5
1	423	155
Range	500.5	450.5
Mean	15.5	2.5

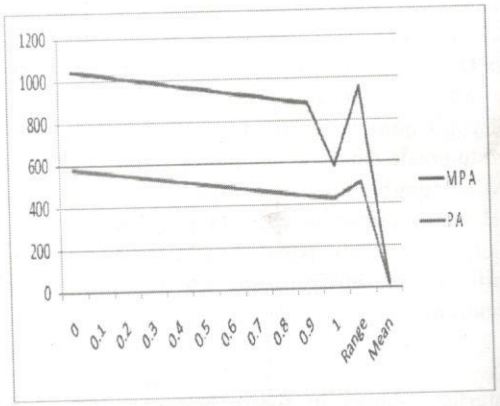


Fig 8: Comparison on network load

Table 2: Comparison on transmission time

Σ	Existing pull all	Modified pull all
0	263.13	147.63
0.1	248.13	145.11
0.2	232.47	142.58
0.3	216.82	140.66
0.4	201.16	137.53
0.5	185.51	135.01
0.6	169.85	132.48
0.7	154.2	129.96
0.8	138.54	127.43
0.9	122.89	124.91
1	107.23	122.38
Range	185.51	135.01
Mean	15.66	2.53

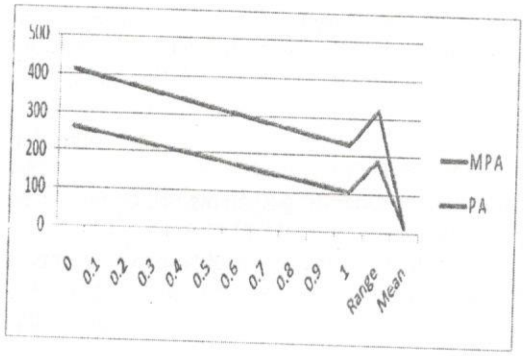


Fig 9: Comparison on transmission time

CONCLUSION

In this paper, we presented a design approach based on modified pull code data migration pattern using Aglets software for sending and retrieving information in a LAN. The implementation was carried out by sending an aglet from the server system to the desired client system using the aglet transfer protocol (ATP) provided in the aglet environment. There was successful dispatching of aglets to their remote hosts for the purpose of returning with the result (data) together with the transmission time and the network load. The aglet returned from the client and displayed the transmission time, network load and results of the client system visited on the server were noted. Based on our results, it is clearly observed that the modified pull code data migration pattern is useful for information retrieval in a distributed environment. The implementation demonstrated time-management and cost-effective alternative compared to the traditional way of retrieving data used today in industries.

FUTURE WORK

The fact that Aglets is now open source and with its powerful architectural framework, researchers could develop more functions into Aglets platform which will make it a tool that could compete favourably well with present network applications. In addition, future work can centre on Network Deployment of other data migration pattern using wide area network.

REFERENCES

Adhicandra, C. Pattinson, E. Shaghouei(2003).Using mobile agents to improve performance of network management operations.ISBN: 1-9025-6009-4 © 2003 PGNet.

Baldi, M. and Picco.G.P. 1998.Evaluating the tradeoffs of mobile code design paradigms in network management applications.*Proceedings of the 1998 International Conference on Software Engineering (ICSE'98), Japan.* R. Kemmerer and K. Futatsugi, eds. IEEE Computer Society Press, 146-155.

Bald, M., S. Gaians G.P. Picco 1997. Exploiting Code mobility in decentralised and flexible network on Mobile Agents(MA), Germany K. Rothermel and R. Popescu-Zeletin (Eds.). Lecture Note in Computer Science, Pub. Springer-verlag, 1219: 27-34.

Braun, P. 2003. The migration process of mobile agents: implementation, classification and optimization. PhD Thesis. Friedrich-Schiller-Universitat Jena.xvii+293.

Braun, P. and Erfurth, C, and Rossaki, W. (2001). Performance evaluation of various migration strategies fo mobile agents. Fachtagung Kommunication in Verteiten Systemen, Germany. Springer-Verlag, 315-324.

Braun, P. and Kern, S. 2005. Towards adaptive migration techniques for mobile agents.Retrieved July 20, 2005 from http://wwwpoleia. lip6.fr/~guessoum/ AAMAS/Finals/braun-kern-aamas-2005.pdf

Braun P. & Whelm, R. (2005) Mobile Agent Basic concept, Mobility Models and the Tracy Toolkit.Morgan Kaufman Publisher.

M. Bernich, F. Mourlin, "Mobile agent communication scheme", International Conference on Systems and Networks Communication (ICSNC '06), pp. 6-6, October 2006.

A. Carzaniga, G.P. Picco and G. Virgna, (1997). Designing diostributed Applications with Mobile Code paradigms. Taylor, R. (Ed.). Proc. 19th Int. Conference Software Eng., pp: 22-32.

C. Erfurth, P. Braun, and W. Rossak (2001) Some thoughts on migration intelligence for mobile agents. Technical Report 09/01. Friedrich-Schiller-Universitat Jena, Institut fur informatik, April 2001

A. Iqbal, J. Baumann, and M. Straber (1998) efficient algorithm to find optimal agent migration strategies.

Technical Report 1998/05, Universitat Stutgart, Fukultat fur infromatik, April 1998.

Gupta R. and Kansal G.(2011). A Survey on Comparative Study of Mobile Agent Platforms International Journal of Engineering Science and Technology (IJEST). Vol. 3 No. 3 March 2011

Gavalas, D., Greenwood, D., Ghanbari, M., and M. O'Mahony, Using Mobile Agents for Distributed Network Performance Management, the 3rd International Workshop on Intelligent Agents for Telecommunication Applications (IATA'99), Stockholm, Sweden, Aug. 1999.

Hunt, C. 2002. *TCP/IP network administration*.3rd edition. O'Reilly & Associates, Inc

H. Kim and N. Feamster "Improving Network Management with Software Defined Networking " IEEE Communications Magazine, February 2013.

L. Xuhui, C. Jiannong, H. Yanxiang, C. Yifeng, MADESE: A Simulation Environment for Mobile Agent, Proceedings of CIT 06, IEEE Computer Society, 2006.

Kona, M.K., and C.Z. Xu, A Framework for Network Management using Mobile Agents, in proceeding of ICEC 2002.

M. K. Kona and C. Xu(2002) A Framework for Network Management using Mobile Agents.

Lange D. B. And Oshima M.(1999) " Seven good reason for Mobile Agents''. Retrieved on 24th March 2012 from Communication of ACM Vol No 3 pp 88-89 March 1999

M. A. Madkour, K. Moria, F. E. Eassa, & Kamal M. Jambi(2014) Mobile Agent Framework for Distributed Network Performance Management. *International Journal of Computer Applications (0975 – 8887) Volume 88 – No.5, February 2014.*

Osunade, O. (2007). Data Migration Patterns for Java Based Mobile Agents. PhD.Thesis, Department of Computer Science, University of Ibadan.

Osunade, O. And Atanda F.A.(2008) Analysis of two Mobile Agents' Data Migration patterns Journal of Mobile Communication 2(2):64 – 72 . Available athttp://www.medwelljournals./new5/archivedetails.php

Osunade, O. R. D. Oloritun, A. Dhabi(2013) Mobile Agent Application Development in a simple java-based mobile agent system(SIMMAS) International Journal of computers & Technology Vol17, No2. Pg 565- 573.

Oyewole, S A., Osunade, O & Azeez, N.A. (2012).A Java Simulation-Based Performance Evaluation of Mobile Agent Platforms.Computing, Information Systems & Development Informatics Journal.Vol 3, No.3. pp 69 -.76 Online at www.cisdijournal.net

Oyewole S. A and Osunade, O.(2011). Effect of data compression on Network-load and Transmossion-Time in Aglet mobile agent. Journal of technology and vocational Educators (JOTTAVE) Vol 2 No. 1 pp 86-93 June 2011 .

Powell, M.L. and Miller, B.P. 1993. Process migration in DEMOS/MP.Proceedings of the 9th ACM Symposium on Operating Systems Principles. ACM Press, 110-119.

Venners, Bill. (1997). Solve real problems with aglets, a type of mobile agent.Retrieved March 12, 2008 from www.javaworld.com/javaworld/jw-05-1997/jw-05-hood.html.

Vigna, G. (1998). Mobile code technologies, paradigms and applications. PhD. Thesis. Politecnico Di Milano, Italy. iii+84.

WJ Buchanan, M Naylor and AV Scott(1999) Enhancing Network Management using Mobile Agents.

H. Zhen, W. Zhen, L. Xiao, Formal Language Description of Mobile Agent's Theory Model, Proceedings of Machine Learning and Cybernetics International Conference ,pp. 185-187, 2006.

L. ZhiLong, C. Hu, and W. Ge. "WBM based network management system design and Implementation" 9th International conference on Fuzzy Systems and Knowledge Discovery (FSKD 2012).