Journal of Engineering and Applied Sciences 5 (2): 84-90, 2010
ISSN: 1816-949X
© Medwell Journals,2010

Improving Availability Using Mobile Agents in Wireless Networking

Michael Hosein and Aleema Khan
Department of Mathematics and Computer Science,
University of the West Indies, St. Augustine, Trinidad

Abstract: The rapid development of the Internet and networking technologies like 802.11n as well as Gigabit Ethernet and FTTH and the related technologies have led to an expansion of shared services and resources. These resources e.g., files can be accessed from desktops, laptops, workstations, pdas, cell phones, televisions in many cases without regard to location and at very good speeds. Mobile agents can play an important role in allowing anytime/anywhere access. This study attempts to investigate the use of mobile agents as a means of improving data availability over a wireless network. Resources were distributed using the static client/server model as well as the dynamic mobile agent model and the results showed the viability of the mobile agents approach.

Key words: Mobile agents, availability, wireless networks, data lockers, pervasive computing, Trinidad

INTRODUCTION

The rapid developments in wired and wireless technologies has helped to facilitate reliable as well as fast access to data. The study investigates the use of mobile agents to improve data availability over a wireless network. This investigation is important in realizing the full potential of mobile agents as being reliable and efficient in the transmission of data across a network and can be applied to the communication of information in wireless networks where the nature of the communicating environment is more unreliable than that of wired networks. It is hoped that this study would also spur more attention that is needed in this area of research of mobile agents as a developing trend and technology. The aims of the study were achieved by allowing mobile agents to be dispatched across a network and to communicate with static agents. The approach was then compared to a static client/server model.

Computer communications using mobile agents showed the ability of mobile agents to move around a network in a much seamless manner compared to other applications comprising of static models, like the client-server architecture. This automatic migration of mobile agents allows them to improve the availability of data in a wireless network. A simulated wireless network was used in the experiments.

Literature review: Mobile agents are program entities that are able to move from one host to another within a computer network. When mobile agents move around, they carry code as well as program state. This saving of the programming state allows the mobile agent to be transported to another location where its execution can continue rather than restart. Mobile agents are suitable for wireless networks since agents can move across unreliable or weakly connected networks and reside elsewhere, possibly on full client nodes.

Many approaches for developing pervasive services have been proposed and mobile software agents are more and more seen as an attractive option (Satyanarayan *et al*., 2001; Cardoso and Kon, 2002). Many mobile agents are written in Java and because of the JVM, they are independent of the operating system or computer architecture of target nodes. Mobile agents move independently from the system but in process-migration systems, running processes are instructed when to move by the system. Mobile agents also differ from applets which are downloaded based on user interaction and run to completion on the host device. Mobile agents have proven to be a better choice for many applications. For one, mobile agents tend to lead to decreases in network latency and bandwidth of client-server applications. Also, they tend to reduce the vulnerability to network disconnection and weak connectivity. However, not all applications will need mobile agents for the entire duration of execution of their tasks. Mobile agents are sometimes useful for only part of a process execution.

There have been a number of researches done in the area of mobile agents. Kutila *et al*. (2009) compare a mobile agent approach using VERSAG with non-mobile agent approaches. Arunachalan and Light (2008) describe

Corresponding Author: Michael Hosein, Department of Mathematics and Computer Science, University of the West Indies, St. Augustine, Trinidad

AMMA, a mobile agent system for reliable clinical data mobile messaging. The various advantages that mobile agents bring to distributed computing scenarios have been extensively discussed in the literature. It is generally accepted that mobile agents produce less network traffic in comparison to the client-server paradigm in certain situations (Picco, 2001; Braun and Rossak, 2004). Mobile agents may prove to be an upcoming technology, however it is not yet perfect. Many researchers are doing the resaerch on developing better methods for improving the technology with better programming environments with more standardization and design patterns. As for security issues such highlighted in Claessens *et al.* (2003), malicious mobile agents may damage local systems or local systems may capture mobile agents and obtain private information. They show cases where many open issues regarding mobile agents and security issues are areas for further research. On the up side of security research, study has been done on mobile agents in secure electronic transactions (Claessens *et al.*, 2003), in patient healthcare within hospitals systems (Arunachalan and Light, 2008) to just name a few.

MATERIALS AND METHODS

Two models were designed. One model used mobile agents where the mobile entities traveled across a simulated wireless network. The other model used a static client/server application over a simulated wireless network. The programming language used (Java) facilitated for the heterogeneity of the network and operating systems. This ensured for the reuse of the same source code for other platforms. Two databases were designed for each prototype using Microsoft Access; these databases held user information as well as recorded diagnostic results as programs were executed.

The experiments looked at various design techniques to improve the availability of data within wireless networks with the use of mobile agent technology.

The availability of data was determined by comparing the mobile and static agent application with the client/server application by obtaining results for the following criteria in each case:

- Bandwidth used that is the amount of data sent in kilobytes per ms (milli sec)
- Time taken for storage of files of various sizes
- The number of transmissions over the network for the storage of files of various sizes

This criteria were calculated within the code for each prototype that is the mobile agent prototype and the client/server prototype developed. Two prototypes were developed, one for a file transfer over a network with the use of mobile and static agents and the second application used a client/server architecture over a network. The latter was used to compare it with the mobile and static based technique. In both cases the times taken for files to be stored over the network were compared. Bandwidth as well as the number of transmissions of data sent for each file transfer over the connection were also compared for both mobile-agent and client-server prototypes.

The Data Locker technique taken from (Villate *et al.*, 2002) was used in the research as an example as to demonstrate how data can be made more available within a wireless network. Therefore, this example was used in order to give a more realistic scenario towards the aims of the research.

The mobile agent program consisted of mobile and static agents. The mobile agents travelled across a simulated wireless network and communicated with static agents for the purpose of file transfers. The mobile agents acted on behalf of the user for file retrieval or storage. These mobile agents also accessed a database on the server side, which was used to store the user's files (data). Figure 1 shows this mobile agent approach.

The client/server implementation consisted of static programs that used the method of request/reply form of communication (Remote Procedure Call (RPC)). This prototype consisted of programs on the client side and programs on the server side, where the client side programs sent or received files from across the network and the server side storing or retrieving files as indicated by the user. The server side program accessed a database consisting of information about the user as well as storing the user's files (data). Figure 2 shows the client/server approach. The platform used to develop the mobile and static agents was the Aglets Software Development Kit (ASDK). This was developed at the IBM research laboratory in Japan. The ASDK provides a framework for

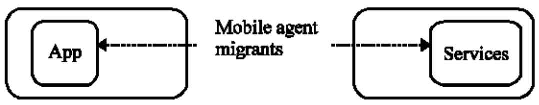


Fig. 1: Mobile agent network load

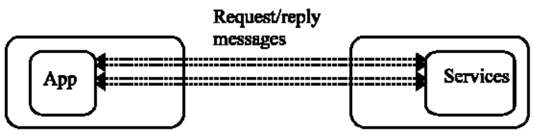
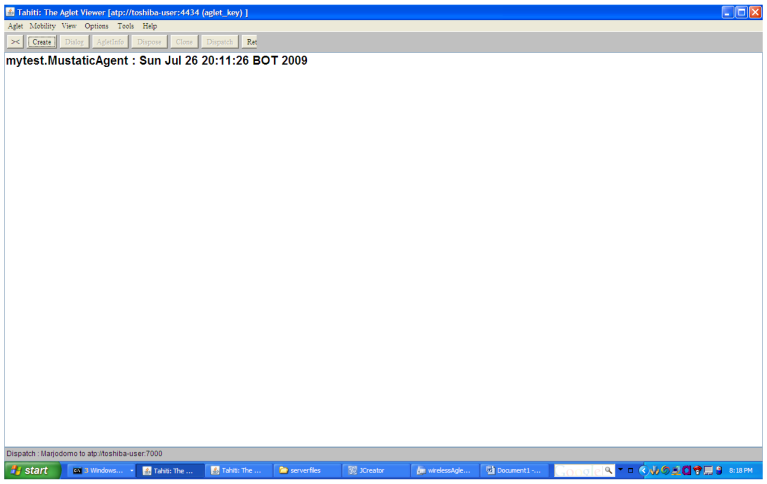


Fig. 2: Client/server network load


Tahiti: The Aglet Viewer [atp://toshiba-user:4434 (aglet_key)]
Aglet Mobility View Options Tools Help
Create Dialog AgletInfo Dispose Clone Dispatch Ret
start

mytest.MustaticAgent : Sun Jul 26 20:11:26 BOT 2009

Dispatch : Marjodomo to atp://toshiba-user:7000

Fig. 3: Agent running on Tahiti

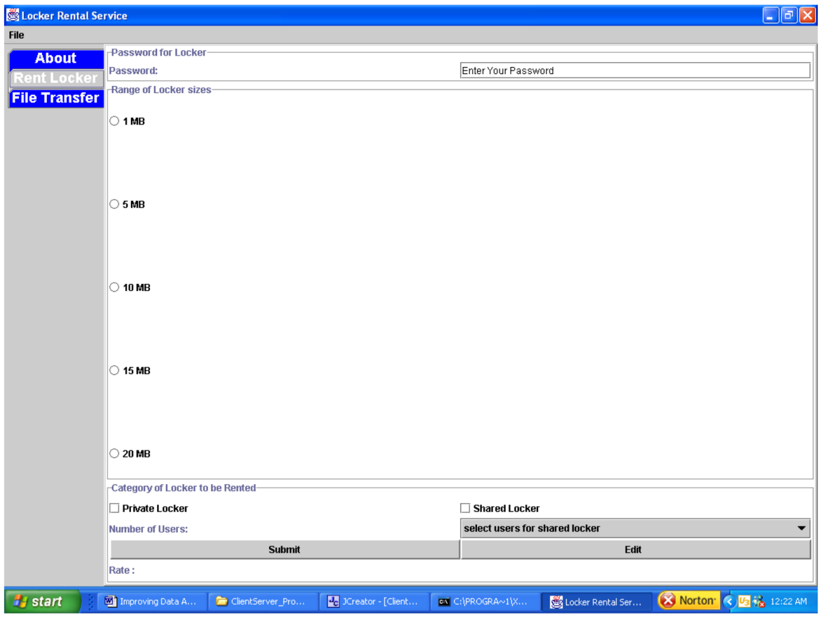


Fig. 4: GUI for renting a Locker

developing mobile agents which includes Tahiti, a visual aglet management and monitoring GUI. Figure 3 shows a screenshot of an agent running on Tahiti. The ASDK is an environment used for the programming of mobile agents in Java. Aglets are the names given to the objects (mobile agents) that move about from host to host. The Aglet can execute, halt its execution on a host, move to another host and resume its execution there. Aglets support strong mobility, this means that the program code as well as the data move with the agent where ever the agent goes. For the case of the static client/server model an interactive GUI was created, the screenshot in Fig. 4 shows one of the option from the GUI. Data was inputted through the GUI and was stored directly to the database.

The client-server prototype consists of the classes:

- Client Driver ( ) gives a Graphical User Interface (GUI) where the user can input data
- Rent Locker ( ) forms the panel for the GUI to accept the user's data to rent a locker
- File Transfer ( ) forms the panel for the GUI to accept user data to either store or retrieve a file
- Mustatic Agent ( ) creates a connection with the Server ( ) class and sends the data through Marjodomo Agent
- Marjodomo Agent ( ) user's request data values sent to the server are given to Majordomo Agent on server side and creates Lockerrent Agent

- Lockerrent Agent ( ) given user's data values from the Marjodomo Agent
- Locker Guardian Agent ( ) queries, updates and inserts into the database with the user's data, times taken for data transfers and bandwidth calculation
- Server creates a connection to the Mustatic Agent

These classes are named similarly to that of the mobile-agent prototype but these classes do not have the features of mobile agents.

The mobile-agents prototype consists of the following classes:

- The MUstaticAgent ( ) resides on the client side, it also creates the Majordomo Agent and passes the data to it and sends it to the server side
- The Majordomo Agent ( ) is the mobile agent used to carry data as well as it's code to the server side
- The Lockerrent Agent ( ) is created by the Majordomo Agent upon arrival on the server side and the data the Majordomo Agent carries is passed to it
- The Lockerguardian Agent ( ) resides on the server side and does the insertion of times taken for a store command, it also copies the file that needs to be stored. Also inserts the bandwidth calculated for each file being sent or received

RESULTS

The tests conducted all aimed to show that data can be made more available with the use of one of the prototypes. This was accomplished by comparing the times taken, bandwidth and number of transmissions across the connection for the file transfer from client to server. The less time taken for file transfers showed which technology proved to be better at transferring the data across the network in less time thus proving itself to be a more efficient method for file transfers. The bandwidth calculated showed how much data was sent for each file per millisecond given differing file sizes, this showed that a great amount of data can be transmitted in less time so the data rate showed higher for the more favourable prototype. The number of transmissions showed how many times data was sent through each prototype's connection for the time taken in the storing of a file, this showed that there was less loss of packets or less packet loss rate for the minimal number of times data is sent for a file storage.

Test 1: Time taken for storing files in mobile-agent and client-server prototypes: For this test, the sending and retrieval of 10 files were done 20 times for each file. These files varied in size from 10-100 kb and the average times were measured for these files to get from each prototype's client to server in the case of file storing. For file storing, the time was calculated using the formula shown where tn is time in milli sec:

$$\text{Average time taken} = (t1 + t2 + t3 + t4 + \ldots .. + t20)/20$$

This formula was calculated within the code for each prototype and stored within each prototype's database.

Time taken for file storing: Table 1 shows the results of the average times taken in both mobile-agent and client-server prototypes for the storing of files. It can be seen that more time is required to store a file using the client-server prototype. With the mobile-agent prototype it takes considerably less time than the client-server prototype to store the same file for every file stored.

Based on the results obtained the following graph shown in Fig. 5 was plotted using the average times in milli seconds for each file varying in sizes from 10-100 kb. As mentioned before, it is clear that the average time to store the files is less for the mobile-agent prototype than with the client-server prototype. The average times for storing of files increased for each prototype as the size of

Table 1: Average times taken for storing of files of various sizes for mobile-agent and client-server prototypes

File size (kb)	Average time in milli seconds (ms) for mobile-agent prototype	Average time in milli seconds (ms) for client-server prototype
10	57.80	132.85
20	72.65	99.65
30	88.25	122.55
40	111.00	183.25
50	146.80	165.45
60	157.05	169.50
70	68.70	163.75
80	136.75	211.40
90	129.85	275.35
100	156.15	230.12

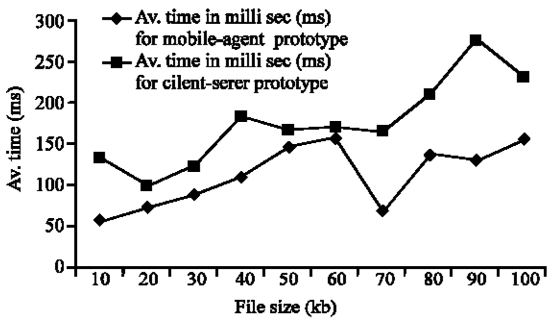


Fig. 5: Graph showing average times taken for storing of files of various sizes for mobile-agent and client-server prototypes

the file also increased. So with larger files more time will be taken in both cases. The results here favour the mobile-agents as opposed to the client-server approach. Despite the fact that mobile agents travel with their code they are still able to carry data with them in a faster time than the client-server approach which relies upon reply/request messaging.

Test 2: Bandwidth used for storing files in mobile-agent and client-server prototypes: For this test the sending and retrieving of 10 files were done 20 times for each file, these files varied in size from 10-100 kb. The average bandwidth for each file to get from each prototype's client to server for file storing was calculated using the formula shown below where bn is number of bytes of data and tn is the time in milli sec. For the average bandwidth calculations for the mobile-agent prototype the size of the mobile agent class was included in bn, where the Majordomo Agent has a size of 2048 bytes, this was done since this agent travels across the connection thereby adding to the bandwidth:

$$\text{Average bandwidth} = (b1/t1 + b2/t2 + b3/t3 + b4/t4 + \ldots.. + b20/t20)/20$$

Bandwidth for file storing: Table 2 shows the results of average bandwidths of both mobile-agent and client-server prototypes. As shown here the mobile-agent prototype maintained a higher bandwidth than the client-server prototype for the storing of files.

Based on the results obtained, the graph shown in Fig. 6 was plotted using the average bandwidth in bytes per milli sec for each file varying in sizes from 10-100 kb. As shown the mobile agent prototype had the greatest number of bytes transmitted per second for file storing than the client-server. This proves that the amount of data transmitted in 1 milli sec for the mobile-agent prototype was greater than that for the client-server. More data is being sent across the network means that files with large sizes such as images, jpegs photos and so on can be transmitted across in less time. This means that more data can be made available in a shorter time using mobile agents, thus proving our data availability issue. However, the downside to this is the bandwidth consumption of the connection, since mobile agents travel with their code as well this adds to the bandwidth not being conserved. Thus mobile agents are better suited for connections with high bandwidths.

Test 3: number of transmissions for storing and retrieving files in mobile-agent and client-server prototypes: For this test the sending and retrieving of 10 files were done 20 times for each prototype, these files varied in size from 10-100 kb. The number of transmissions of data made for the files to get from each prototype's client to server for file storing was recorded by the code. The formula used to determine the average number of transmissions made for each file is as follows where trn represents the number of transmissions for a file:

$$\text{Average no. of transmissions} = (tr1 + tr2 + tr3 + tr4 + \ldots\ldots.. + tr20)/20$$

Number of transmissions for file storing: Table 3 shows the results of the average number of transmissions of data made in both mobile-agent and client-server prototypes for the storing of files. As can be seen the number of data transmissions for the mobile-agent prototype is shown to be a constant value of 1. However, for the client-server prototype the number of transmissions increased steadily as the size of the file increased.

Based on the results obtained, the graph shown in Fig. 7 was plotted using the average number of transmissions of data in bytes per milli second made for

Table 2 Average bandwidths for storing files of various sizes for mobile-agent and client-server prototypes

File size (kb)	Average bandwidth used in milli seconds (bytes/ms) for mobile-agent prototype	Average bandwidth used in milliseconds (bytes/ms) for client-server prototype
10	242.60	104.00
20	355.90	272.80
30	388.20	308.50
40	440.70	270.50
50	409.55	452.30
60	447.30	463.55
70	1229.20	636.25
80	650.90	495.70
90	784.65	385.65
100	705.35	651.25

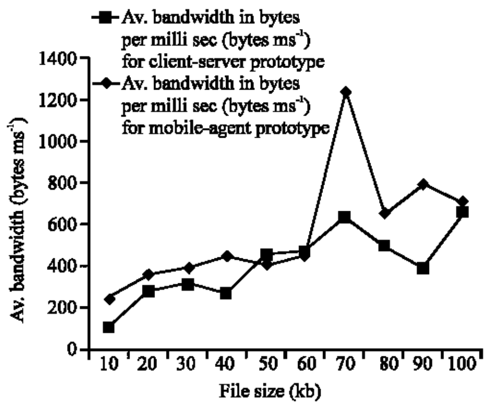


Fig. 6: Graph showing average bandwidth for storing of files of various sizes for mobile-agent and client-server prototypes

Table 3: Average no. of transmissions of data for storing files of various sizes for mobile-agent and client-server prototypes

File size (kb)	Average no. of transmissions of data for mobile-agent prototype	Average no. of transmissions of data for client-server prototype
10	1	10
20	1	20
30	1	30
40	1	40
50	1	50
60	1	60
70	1	70
80	1	80
90	1	90
100	1	100

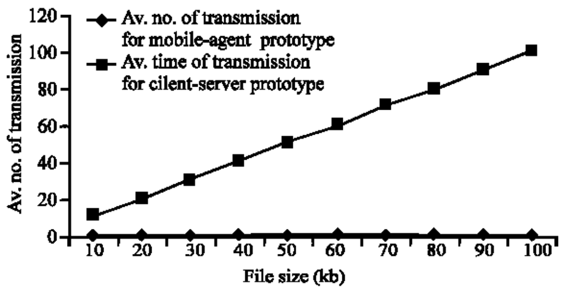


Fig. 7: Graph showing average number of transmissions of data for storing files of various sizes for mobile-agent and client-server prototypes

each file varying in sizes from 10-100 kb. It can be seen on the graph that the number of transmissions for the mobile-agent prototype for storing files was steady and significantly less than the client-server prototype. This proves that the connection for the mobile-agent prototype was only used once to store the file compared to the client-server prototype that used the connection more times as the file size increased. This shows that the communication load for the mobile-agent was less than that for the client-server thereby reducing network traffic considerably. Mobile agents can be used quite efficiently in getting data across a network, by using less of the communication protocol.

DISCUSSION

This research supports the idea that mobile agents could be lead to enhanced performance of network services like file storage. The deployment of the mobile agent across a simulated wireless network gave results that were favourable for data being made more available than the traditional client-server method.

An observation in the context can be made that data availability can be improved for wireless networks with the use of mobile agents. The tests results obtained showed that mobile agents are quite efficient at transferring data across a network. It was also shown that the migration time or the time taken to store data across a network was significantly less than that for the client-server approach. The bandwidth of the transmission for the mobile-agent approach was far greater than that for the client-server approach showing that more data can be sent across the network in less time.

The number of transmissions for the mobile-agent approach was the same for all sizes of files sent as compared to the client-server approach where the number increased as the file size increased. Therefore, network load is decreased for the mobile-agent approach as compared to the client-server approach of multiple interactions to accomplish a given task.

Given all the above analysis of the results it is clear that data is indeed made better available across networks with the use of mobile agents.

CONCLUSION

Mobile agents have proven to be worthy when it comes to data availability. However, more work needs to be done in investigating how efficiently mobile agents and the traditional RPC method of the client/server could co-operate or go hand in hand in delivering quality service to the wireless world. The study provides a starting point for such research.

REFERENCES

Arunachalan, B. and J. Light, 2008. Agent-based mobile middleware architecture (AMMA) for patient-care clinical data messaging using wireless networks. Proceedings of the 12th IEEE/ACM International Symposium on Distributed Simulation and Real-Time Applications, Oct. 27-29, Washington, DC, USA., pp: 326-329.

Braun, P. and W.R. Rossak, 2004. Mobile Agents Basic Concepts, Mobility Models and the Tracy Toolkit. Morgan Kaufmann Publishers, San Francisco, CA., ISBN-10: 1558608176.

Cardoso, R.S. and F. Kon, 2002. Mobile agents: A key for effective pervasive computing. Proceedings of the ACM OOPSLA 2002 Workshop on Pervasive Computing, Seattle, November 2002. British Columbia, Canada. http://www.ime.usp.br/~speicys/publications/oopsla2002.pdf.

Claessens, J., B. Preneel and J. Vandewalle, 2003. How can mobile agents do secure electronic transactions on untrusted hosts? A survey of the security issues and the current solutions. ACM Trans. Internet Technol., 3: 28-48.

Kutila, G. S. Krishnaswamy, S.W. Loke and A. Zaslavsky, 2009. Runtime efficiency of adaptive mobile software agents in pervasive computing environments. Proceedings of the 2009 International Conference on Pervasive Services, July 13-17, London, United Kingdom, pp: 123-132.

Picco, G.P., 2001. Mobile agents: An introduction. Microprocessors Microsyst., 25: 65-74.

Satyanarayan, M., 2001. Pervasive computing: Vision and challenges. IEEE Personal Commun., 8: 10-17.

Villate, Y., A. Illarramendi and E. Pitoura, 2002. Keep your data safe and available while roaming. Mobile Networks Appl., 7: 315-328.