Migration of Mobile Agents in Ad-hoc, Wireless Networks

Migas N, Buchanan WJ and McArtney K
School of Computing, Napier University, EH10 5DT, Scotland, UK
n.migas, w.buchanan, k.mcartney{@napier.ac.uk}

Abstract

This paper focuses on the design and development of a novel architecture called MARIAN, which utilises static agents, mobile agents, and also a hybrid approach, in order to perform routing, network discovery, and automatic network reconfiguration, in wireless ad-hoc networks. The paper shows that, in most cases, the static agent approach is faster than the mobile agent approach in retrieving data from a wireless remote database. However, if the amount of data to be retrieved is relatively large, such as in the gathering of data for routing information, the mobile agents are more capable of filtering data according to the required preferences. It also shows that the time taken to gather routing information can be significantly reduced using a mobile agent approach, as compared with the static agent approach.

1. Introduction

An ad-hoc network consists of mobile devices that have no central administration, and thus form a temporary network. It may therefore be necessary for one mobile device to seek the aid of others in forwarding data packets to their destination, due to the limited propagation range of the device's wireless transmissions [1]. Ad-hoc networks have a vast number of applications, such as in military operations, commercial, disaster relief, conferencing, sensor networks, personal area networks, and embedded computing applications [2].

Routing on mobile devices in wireless, ad-hoc networks is a complex process due to factors including mobility, limitations in processing power and reduced battery capacity. Current routing protocols can be grouped into two categories: proactive and reactive.

- Proactive. These maintain a route to all nodes within the network, including those to which no packets are sent. They also react to dynamic topology changes, even if these changes have no effect on the traffic. Traditional network routing protocols like Destination Sequenced Distance Vector (DSDV) [3] are proactive.
- Reactive: these only react when a route is needed between a source and a destination node, and do not need to try and maintain routes to destinations that they are not communicating with. This includes routing protocols such as Dynamic Source Routing (DSR) [4], Ad-Hoc On-demand Distance Vector (AODV) [5], Ad-hoc On Demand Multipath Distance Vector (AOMDV) [6].

The mobile agent paradigm is a relatively new technology that has its origins in intelligent agents, and is proposed as an alternative approach to client-server communications model. A mobile agent is a software entity that inherits some of the features of an intelligent agent and requires an agent environment to execute. A mobile agent can suspend its execution on a host computer, and then transfer its code, data state, and possibly its execution state (strong migration) to another host on the network that must provide an agent environment, and resume execution on the new host. The aim of an agent environment [20] is to provide the appropriate functionality to mobile agents to execute, communicate, migrate, and use system resources in a secure way. In general, a mobile agent comprises of an agent model, a life-cycle model, a computation model, a security model, a communication model, and finally a navigation model [7].

This paper presents the migration process of mobile agents in wireless ad-hoc networks were participating nodes are mainly mobile devices, such as laptops and PDAs. Section 2 presents background information on Grasshopper micro-edition, which is a mobile agent system capable of running in small Java-enabled devices such as PDAs. Section 3 MARIAN, a potential routing architecture which utilises static and mobile agents to perform routing, network discovery, and automatic network reconfiguration in wireless ad-hoc networks. This is then built on, to present a database application scenario in Section 4, which justifies MARIAN's architecture. Section 5, presents innovative experimental results that further prove the applicability of agents in wireless ad-hoc networks [8-12]. Section 6, concludes this research work.

2. Grasshopper micro edition for PDAs

Grasshopper is a Java-based mobile agent system developed by IKV++ [13]. It builds on top of a distributed processing environment and thus allows the integration of the traditional client/server paradigm, and mobile agent technology [14]. It is compliant with the agent standard defined by the Object Management Group (OMG), which is the Interoperability Facility (MASIF) [15].

It also supports multiple communication protocols, such as Remote Method Invocation (RMI), RMI SSL, Plain Socket, Plain Socket/SSL, and IIOP. Supported communication modes include synchronous, asynchronous, dynamic, and multicast. The unique feature of Grasshopper, and most important for this research perspective, is that it can be executed on small wireless devices such as PDAs, as long as they are, at a minimum, Java 2 Micro Edition (J2ME)-enabled [16].

Grasshopper is an open source project, and, from our experience in using Grasshopper micro edition, there are a number of problems when executing software in PDAs that are J2ME compliant [17]. When the graphical user interface components are turned on, Grasshopper halts execution and prints out a number of exceptions. This may be due to the fact that Grasshopper's graphical components have been developed according to Swing libraries [18], while J2ME only supports AWT 1.1 [19]. This can be fixed by manually disabling the graphical components and work with the provided textual interface, which effectively serves the same purpose.

In Grasshopper, the execution environment of static and mobile agents is called an agency, which can be subdivided in more than one place. In each agent-enabled host, a running agency is necessary in order to execute agents, and also provides services such as communication, registration, management, transport, security, and persistence. Each agency is aware of all currently hosted agents and places for management purposes, by the use of a registration service. Besides the registration service, Grasshopper offers a region registry, which maintains information on agents, agencies, and places in the scope of a whole region. Thus, an agent may ask the region registry for the location of a particular service, and thus migrate there in order to benefit from local interactions.

3. MARIAN routing protocol

Mobile Agents for Routing in Ad-hoc Networks (MARIAN) is a research project that proposes to assess different models of the usage of static and mobile agents to determine the best route through ad-hoc networks [23, 24]. The routing process in wireless ad-hoc networks is a complex one and requires research into the best metrics to identify the best path, such as memory capacity, network performance, processing capabilities, cost, and so on. One model is to use a mixture of mobile and static agents to gather relevant information. These agents could perform important tests, which could be used to generate the best route through a network. This research looks at different models for the deployment of these agents, which balance the usage of static and mobile agents. A number of novelties are expected to emerge that will improve current routing protocols. These include optimisation of network performance, scalability, improved Quality of Service (QoS), reconfigurability, and security.

The research effort has defined a general framework that uses a mixture of static and mobile agents for routing, network discovery, and automatic network reconfiguration in wireless ad-hoc networks. It uses a static agent, a mobile agent, and a hybrid agent approach, in order to be suitable for a vast set of applications. Thus, according to the application's needs, the static, mobile, or hybrid approach can be utilised. There are a number of research projects that currently utilise the mobile agent technology to perform network discovery, and routing in ad-hoc networks. Chpudhury proposed a distributed mechanism for topology discovery in ad-hoc wireless networks using mobile agents [26]. Along the same direction, Marwaha proposed a combination of the on-demand routing protocol Ad-Hoc On Demand Distance Vector (AODV) [27] with a distributed topology discovery mechanism using ant-like mobile agents [28]. Their results, further support the thesis of this research that mobile agent technology can be used for routing, network discovery, and automatic network reconfiguration, in an efficient, effective, and secure way. An aim of MARIAN is to prove that mobile agents can be migrated from small wireless devices using IEEE802.11b standard, and that they provide a better solution to client-server approach when filtering of data is used locally.

MARIAN's architecture is based on the framework developed by the authors of this paper and presented in [25]. As a synopsis, the routing protocol groups mobile devices into wireless domains. The principle is that devices that are situated in the same wireless domain are in direct communication range with each other. A mobile device may belong to one or more wireless domains, and thus a cluster is defined. A cluster is composed of more than one wireless domain, where at least one device belongs to all domains. Once the organisation of devices into wireless domains and clusters is completed, MARIAN chooses the strongest device of each wireless domain to implement a region registry. Every other device in the same wireless domain then registers with the region registry. The device that implements the region registry has knowledge of all other devices situated in the same wireless domain, all agencies and places running in each device, all static or mobile agents, and all services that may be offered by mobile devices.

Even though a mobile agent may migrate from one device to another, the registry maintains a track to the agent's current location, so that communication is handled transparently. When a registered device moves away from the current wireless domain, the region registry erases all references to that device, and informs the rest of the devices that this particular device is no longer reachable.

Figure 1, illustrates the organisation of mobile devices into wireless domains and clusters.

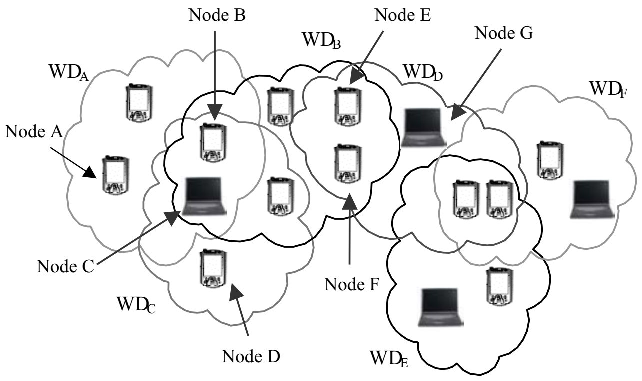

Figure 1: MARIAN, organisation of mobile nodes into clusters and domains

4. Application Scenario

An application scenario has been designed and implemented in order to prove that migration of agents in wireless ad-hoc networks consisting of PDAs and laptops, using the IEEE802.11b standard for communications, is achievable. The application also provides evidence that the architecture of MARIAN routing protocol is feasible and that it may be a better approach to traditional routing protocols by providing a set of advantages such as maximise network performance, scalability, dynamic, Quality of Service (QoS), reconfigurability, and security.

Figure 2 illustrates four wireless devices, in which two of them are PDAs and the other two are laptops. One of the laptops maintains a public database of articles from journals, conferences, workshops, and tutorials. It also provides a simple search facility that once a query is passed, it returns a number of hits that include the article's unique identification number, authors, summary, and so on. A client PDAs wants to search laptop's public database, however, it is not in direct communication range. Fortunately, the other devices (PDA and laptop) are situated in between the client PDA and the database laptop. Thus they are in direct communication range with the client PDA, the database laptop, and one another.

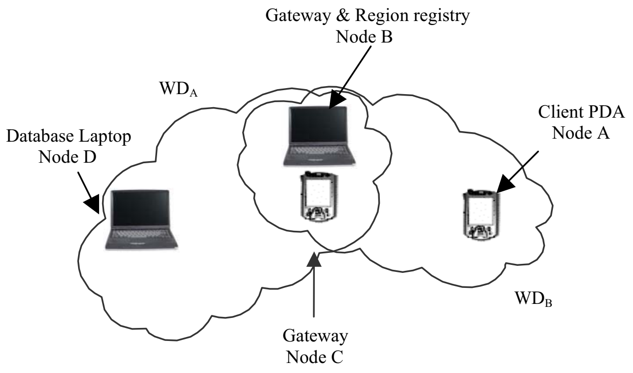

Figure 2: Organisation of devices for the database application scenario

In this case two wireless domains are identified (WDA and WD_B) and one cluster (C_A ($WD_A \cup WD_B$)), which consists of both wireless domains (WD_A and WD_B). The two PDAs, node B and C belong to both wireless domains, and therefore can both act as bridges in order to link both wireless domains together. In this way, node A will be able to communicate with node D, through either node B or C. We can assume that the strongest device of both wireless domains (WD_A and WD_B) is node B. Therefore node B was chosen to implement a region registry. The application is thus designed in such a way that node A asks the registry if there is a database service available. The registry replays back with a positive answer and provides the location and route to contact the database laptop, which, in this case, is either $A \rightarrow B \rightarrow D$ (route 1) or $A \rightarrow C \rightarrow D$ (route 2). By default, the application selects the second route, which is through the gateway PDA. Even though, MARIAN would select the route consisting of the strongest devices (route 1), since both routes require the same number of hops, the application selects the weakest one (route 2) for testing purposes.

The application scenario implements both a static and mobile agent approach. Figure 3, illustrates the static agent approach. Once the route is retrieved, a static agent on node A transmits the query to the gateway agent on node C, which then forwards it to the database static agent on node D. The database agent processes the query and then passes back the results to the gateway agent on node C. Finally, an agent on node C passes the results back to agent on node A.

According to the mobile agent approach, once the route is retrieved, a mobile agent is created having set its itinerary to the retrieved route and carries in its payload the query string. The agent is then serialised (code and data state) and transmitted to the first hop of its destination, which is the gateway node. A new instance of the client mobile agent is then created by the mobile agent system of the gateway node. The mobile agent then requests its migration to the next hop, which is the database node. The agent is then serialised and transmitted to the database node by the gateway's mobile agent system. A new instance of the agent is then created to the database node. The agent then senses its arrival on the database node and initiates communication with the database agent. It passes the query to the database agent and stores the results in its payload. After this it then asks the mobile agent system to transmit it to the gateway node. Upon arrival at the gateway node, the client agent requests its transmission back home (client device). The gateway's node mobile agent system serialises the agent and transmits it back home. Figure 4 illustrates the solution according to the mobile agent approach.

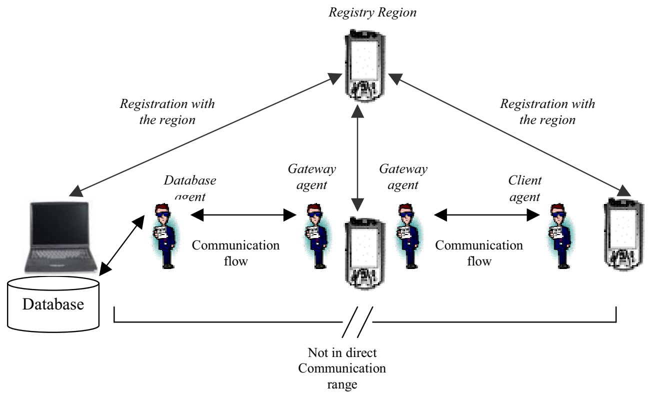

Figure 3: Accessing the database, static agent

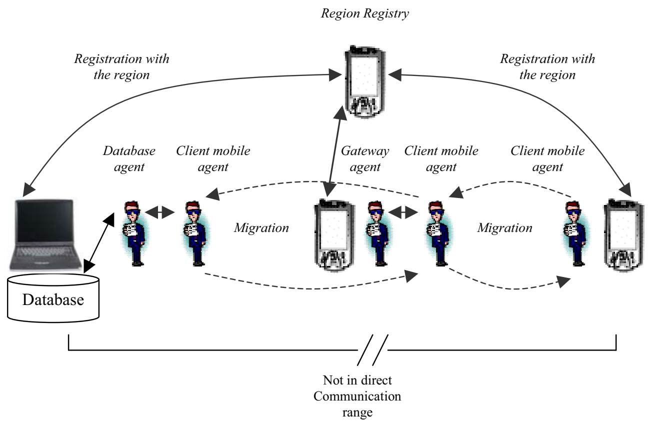

Figure 4: Accessing the database, mobile agent

In addition to the static and mobile agent models, a filtering mobile agent model has also been implemented and tested. The filtering mobile agent model follows the same principles as the one without filtering, however, the client mobile agent maintains preference information on articles that its user is most interesting in. Once the client mobile agent, retrieves the results from the database, instead of just storing them to its payload, it first filters the data locally according to its user's preference, and thus stores only a small amount of the total results. For instance, according to the current implementation of the application scenario, the agent knows that its user is only interesting in recent articles from journals only, written by a set of authors, which their keywords match with the keywords supplied. The agent interprets the word "recent" to papers written between years 2002 and 2003, and considers articles that were published in journals only by a list of specific authors with keywords that match the supplied ones.

5. Results

Initially, the results from the database were set to be 100KBits in size which then increased to 200KBits and finally to 300KBits. Both static and mobile agent approaches were tested against the amounts mentioned above in respect to time. Thus, the time it takes for the client to contact the region, retrieve the route, and get the results from the database was measured. All experiments were iterated 20 times. Figure 5, illustrates the time taken for the client to retrieve 100KBits, 200KBits, and 300KBits according to static agent approach. The horizontal axis represents the iterations that took place (in this case, 20) while the vertical axis represents the time measured in seconds to complete the process.

Figure 6, illustrates the time taken for the client to retrieve 100KBits, 200KBits, and 300KBits according to mobile agent approach. The horizontal axis represents the iterations that took place (in this case, again, it is 20) while the vertical axis represents the time measured in seconds to complete the process. Different sizes have a noticeable effect on time. As expected, size of 100Kbits achieves the best time with an average of 44 seconds, while size of 200Kbits achieves an average of 50.5 seconds and size of 300Kbits achieves an average of 63.5 seconds. Therefore, there is approximately an increase of 15 seconds for an added size of 100Kbits. In the first iteration of the data size 100Kbits and 200Kbit there is a glitch, which as mentioned above, may be caused by either the JVM, or the operating system.

Figure 7, presents the average times of the static and mobile agent approach against 100Kbits, 200Kbits, and 300Kbits sizes of data. It thus contrasts the static and mobile agent approach to retrieve data of size 100Kbits, 200Kbits, and 300Kbits. It can seen that the static agent approach performs significantly better in respect to time. The overall average time for the static agent approach is approximately five seconds while the average time for the mobile agent approach is approximately 50 seconds.

Therefore, the static agent approach performs nearly 10 times better than the mobile agent approach. The delay in the mobile agent approach is based on the fact that the JVM needs to serialise the mobile agent (code and state) in order to transmit it, deserialise it to the destination, and then create a new instance of the agent at the destination node. In the application scenario, this process happens four times, exactly as the migrations of the mobile agent. Thus, the overhead in the mobile agent approach is approximately 45 seconds.

Figure 8 illustrates the time taken to retrieve data of 15Mbits based on the filtering mobile agent approach and contrasts the results to the time taken to retrieve the same amount of data (15Mbits) from the static approach. This shows a significant improvement of the filtering mobile agent approach in contrast to the static agent approach. The average time taken to retrieve data of size 15Mbits, based on the static agent approach, is approximately 62 seconds while the average time taken to retrieve the same amount of data based on the filtering mobile agent approach is approximately 42 seconds..

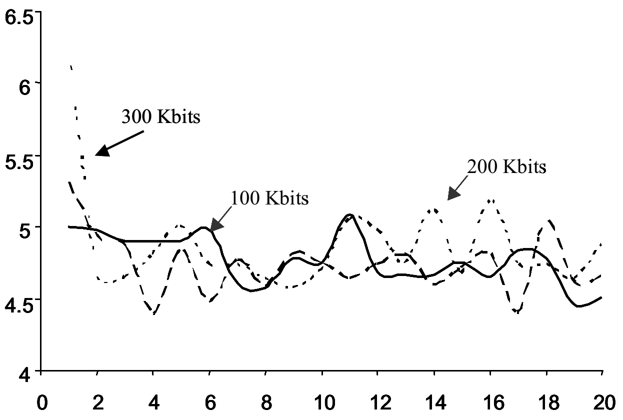

Figure 5: Time taken to retrieve 100KBits, 200KBits, and 300Kbits according to the static agent approach

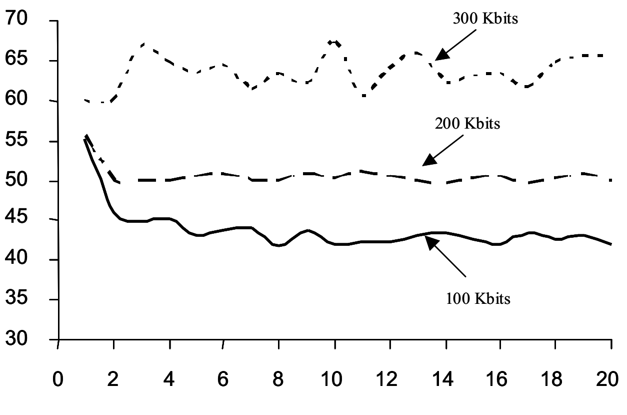

Figure 6: Time taken to retrieve 100KBits, 200KBits, and 300Kbits according to the mobile agent approach

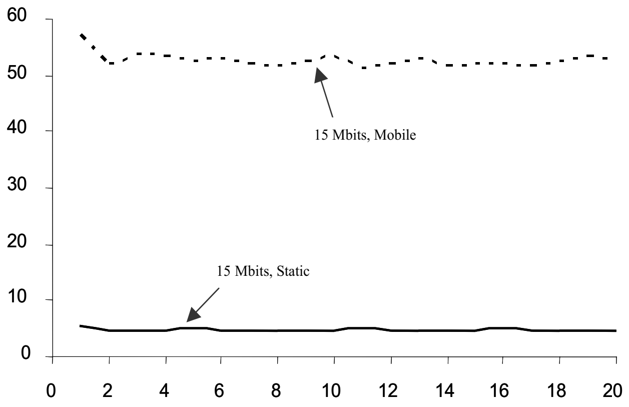

Figure 7: Time taken to retrieve 100KBits, 200KBits, and 300Kbits according to the mobile agent approach

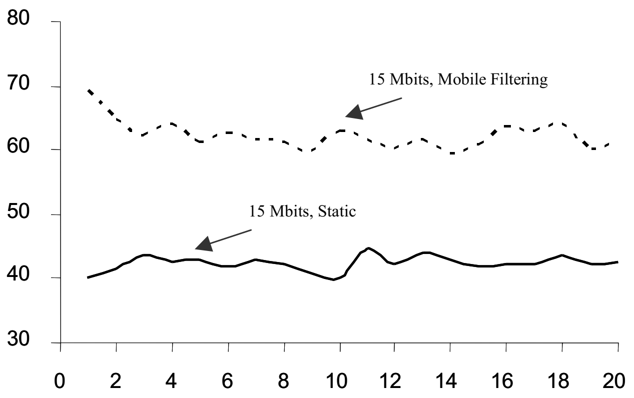

Figure 8: Average times to retrieve 100KBits, 200KBits, and 300Kbits of data

Therefore, the filtering approach is nearly one third faster than the static approach. This is due to the fact that the mobile agent retrieves the data from the database, which are of size 15Mbits, and performs filtering of data locally according to its user's preferences, which, in this case, reduces the data from 15Mbits to only 56Kbits, which then stores to its payload. The static agent approach has no filtering capabilities, and thus retrieves the full amount of data

6. Conclusions

This paper provides background information on wireless ad-hoc networks, traditional and innovative routing protocols, and mobile agent technology. It introduces Grasshopper mobile agent system, which is currently the only system that is capable of executing to small wireless devices such as PDAs, although, problems may arise when grasshopper executes on J2ME-enabled devices. A database application scenario has been designed and implemented according to MARIAN's architecture and experimental results prove that migration of agents in wireless ad-hoc networks consisting of PDAs and laptops, using the IEEE802.11b standard for communications, is achievable, and can be a better approach to static only if local filtering of data is supported. Also, results proved that the proposed architecture of MARIAN routing protocol is feasible and that is may be a better approach to traditional routing protocols, since it utilises both a static and mobile agent approach to suit a vast majority of applications.

7. References

[1] Hassanein, H. and Zhou, A., 2001. Routing with load balancing in wireless Ad-Hoc Networks. Proceedings of the 4th ACM International Workshop on Modelling, Analysis, and Simulation of Wireless and Mobile Systems. Rome, Italy. pp. 89-96.

[2] Perkins, C. E., 2001. Ad-hoc networking: an introduction. Ad-hoc networking. Published by Addison-Wesley. ISBN: 0201309769.

[3] Perkins C. and Bhagwat, P, 1994. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. Proceedings of the ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications. London, UK. pp. 234-244.

[4] Johnson D. B., et. al., 2002. The Dynamic Source Routing protocol for Mobile Ad Hoc Networks (DSR). Internet Engineering Task Force (IETF), Mobile Ad Hoc Networking working group (MANET) Official Internet Draft. http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-07.txt.

[5] Perkins C, et. al., 2002. Ad-hoc On-demand Distance Vector Routing. Internet Engineering Task Force (IETF), Mobile Ad Hoc Networking working group (MANET) Official Internet Draft. http://www.ietf.org/internet-drafts/draft-ietf-manet-aodv-12.txt.

[6] Marina, M. K. and Das, S. R., 2001. On-demand Multipath Distance Vector Routing for Ad Hoc Networks. In Proceedings of the International Conference for Network Protocols (ICNP). Riverside, USA. pp. 14-23.

[7] Harrison, C. G. et al, 1995. Mobile Agents: Are they a good idea. Technical Report. IBM T.J. Watson Research Centre.

[8] Baumann, J. et. al., 1998. Mole concepts of a mobile agent system. World Wide Web. Vol. 1, No. 3. pp.123-37.

[9] Tcl, Developer, 2003. Tcl language. http:// www.tcl.tk/.

[10] Gosling, J. et. al., 2000. The Java Language Specification, 2nd edition. The Java Series. Published by Addison-Wesley. USA. ISBN: 0201310082.

[11] Lindholm, T. and Yellin, F., 1999. The Java Virtual Machine Specification, 2nd edition. The Java Series. Published by Addison-Wesley. USA. ISBN: 020163452X.

[12] Funfrocken, S., 1998. Transparent Migration of Java-Based Mobile Agents. Proceedings of the Second International Workshop on Mobile Agents (MA'98). Stuttgart, Germany. pp. 26-37.

[13] IKV++, Inc., 2003. Grasshopper mobile agent system. Grasshopper Documentation. http://www.grasshopper.de.

[14] Grasshopper Documentation, 2001. Basics and Concepts 2.2. http://www.grasshopper.de

[15] OMG, MASIF, 1997. Mobile Agent Facility Specification. Object Management Group (OMG). http://www.omg.org/docs/orbos/97-10-05.pdf.

[16] Sun, Microsystems, 2003c. JavaTM 2 Micro Edition (J2METM). Sun Community Source Licensing (SCSL). http://wwws.sun.com/software/communitysource /j2me.

[17] Sun Microsystems, 2003a. Java™ 2 Platform, Micro Edition. http://java.sun.com/j2me/j2me-ds.pdf.

[18] M. Robinson and P. Vorobiev, 2000. Swing introduction. http://developer.java.sun.com/ developer/Books/swing2/.

[19] Sun Microsystems, inc, 2002b. Abstract Window Toolkit (AWT). http://java.sun.com/j2se/1.4.1/ docs /guide/awt/.

[20] Silva, A. R. et. al., 2001. Towards a reference model for surveying mobile agent systems. Autonomous Agents and Multi Agent Systems. Vol. 4, No. 3. pp.187-231.

[21] Hadjiefthymiades, S. et. al., 2002. Supporting the WWW in wireless communications through mobile agents. Mobile Networks & Applications. Vol. 7, No. 4. pp. 305-313.

[22] Braun, P., 2002. The migration process for Mobile Agents, Implementation, Classification, and Optimisation. PhD thesis. Fiedrich Schiller University of Jena. Germany.

[23] Samaras, G. and Panayiotou, C., 2002. Personalized Portals for the Wireless User based on Mobile Agents. International Conference on Mobile Computing and Networking, Proceedings of the second International Workshop on Mobile Commerce. Atlanta, USA. pp. 70-74.

[24] Migas N. et al, 2003a. Mobile Agents for Routing, Topology Discovery, and Automatic Network Reconfiguration in Ad-Hoc Networks. 10th IEEE International Conference and Workshop on the Engineering of Computer Based Systems. Huntsville, USA, pp. 200-206.

[25] Migas N. et al, 2003b. MARIAN: A Framework using Mobile Agents for Routing in Ad-hoc Networks. IADIS International Conference on WWW/Internet, Algarve, Portugal.

[26] Chpudhury, R. R. et. al., 2000. A distributed mechanism for topology discovery in ad-hoc networks using mobile agents. Proceedings of 1st Annual Workshop on Mobile Ad-Hoc Networking Computing, MobiHOC Mobile Ad-Hoc Networking and Computing. Boston, USA.

[27] Perkins, C. E. and Royer, E. M., 1999. Ad-hoc On-Demand Distance Vector Routing. Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications. New Orleans, USA. pp. 90-100.

[28] Marwaha, S., 2002. Mobile Agents based Routing Protocol for Mobile Ad-Hoc Networks. In Proceedings of IEEE Global Telecommunications Conference (GLOBECOM'02). Taipei, Taiwan. pp. 17-21.