A Mobile Agent Platform for Supporting Ad-hoc Network Environment*

Jinbae Park, Hyunsang Youn, Eunseok Lee
School of Information and Communication Engineering Sungkyunkwan University
300 Chunchun Jangahn Suwon, 400-746, KOREA
{wingdeng, wizehack, eslee}@ece.skku.ac.kr

Abstract. Networks are becoming much more complicated in their support of ubiquitous computing. For example, broadband mobile communications, wireless ad-hoc networks, IPv6, and so on. Technology is moving towards mobile ad-hoc networks (MANETs), which create temporary networks. The use of mobile devices for composing MANETs is growing, and it is necessary to provide agent service for them. In this paper, we propose a mobile agent platform for supporting MANETs. The proposed algorithms provides agent service among mobile devices, choose an AMS and a DF Service Provider (ADSP) dynamically, maintain the ADSP through another device, and route packets between agents that are unable to communicate. We implement a prototype of these algorithms, based on the Bluetooth protocol. We verify efficient agent management on MANETs, which is not supported by existing mobile agent platforms.

Keywords: agent, agent platform, Bluetooth, ad-hoc, FIPA.

1 Introduction

As information technology is moving towards ubiquitous networks, public interest in MANETs is growing. Using own mobile devices, people easily exchange data each other when they want without wire on the ad-hoc network. Ad-hoc networks don't need any infrastructure to maintain, and consists of voluntary nodes.

An important element of composing MANETs is mobile devices. As uses of mobile devices grow, people use mobile data services more and more, accordingly, data exchange and distributed environment support are becoming an important issue. As a result, agent technology is needed to support process automation, high-level communication and intelligent service on the mobile environment.

The Foundation for Intelligent Physical Agents (FIPA)[1] is a foundation that creates the specifications for agent and multi agent systems. It suggests the use of an Agent Management System (AMS) and a Directory Facilitator (DF), which provide

* This work was supported by the MKE 21st Century Frontier R&D Program in Korea and a result of subproject UCN 08B3-B1-10M, ITRC IITA-2008-(C1090-080-0046), Grant No. R01-2006-000-10954-0, Basic Research Program of the Korea Science & Engineering Foundation.

white and yellow pages service for managing multi-agent systems. In MANETs, the AMS and the DF both have to exist in order to operate agent systems efficiently. In addition, it is necessary to manage them and support several features, depending on the characteristics (dynamic composition, frequent node mobility, etc) of the MANET.

Most of the agent platforms are not implemented to run on MANETs without fixed infrastructure, because hardware on mobile device is constrained, making it difficult to support the agent system perfectly. It is reasonable to assume that an agent platform on MANETs will be possible with the advance of hardware. Currently, it is difficult to implement a complete agent platform because of the limitations of MANETs. However, the agent platform can run at a basic level because MANETs are composed of a small number of devices in most cases.

In this paper, we use the Bluetooth protocol to compose MANETs because it is the most widely used protocol for MANETs. However, in the future, our algorithms will be able to be applied to any other ad-hoc protocol. This paper is structured as follows: Section 2 introduces the FIPA specification, some existing agent systems that support mobile environments and the Bluetooth protocol. The proposed system is described in detail in Section 3. Section 4 presents implementation and experiments to evaluate our algorithms. Finally, conclusion and future works are discussed in Section 5.

2 Related Works

2.1 AMS and DF

The AMS and the DF are agents responsible for agent management service, in the FIPA specification [1]. The AMS is a mandatory element of the agent platform. It will exist only once on a single agent platform. It registers and deregisters agents on the Agent ID directory. In addition, it manages agent migrations on an inter-platform when it is supported. The DF is the component that provides yellow pages service to other agents. Each agent platform must have at least one DF. Multiple DFs may exist within an agent platform and may be federated.

2.2 JADE-LEAP

The Java Agent DEvelopment Framework (JADE) is a software framework implemented in Java. The purpose of the JADE is to simplify multi agent system implementation, compatible with the FIPA specification.[2] To use it on mobile devices, the JADE-Lightweight Extensible Agent Platform(JADE-LEAP) is developed. JADE-LEAP enables agents to be executed on lightweight devices such as cell phones. It is developed to run on mobile device supporting sufficient resources and processing power without any modification and it uses wireless networks [3]

The JADE-LEAP is one of the most popular mobile agent systems. However, it cannot be used where TCP/IP is not supported, and there must be a main container based on J2SE. In other words, it cannot support ad-hoc networks.

2.3 Agent Network for Bluetooth Devices (ANBD)

The ANBD [4] is a system that provides agent service on personal mobile devices using the Bluetooth protocol. ANBD is composed of mobile and fixed devices that are Bluetooth-enabled and equipped with the J2ME execution environment and, fixed devices, such as PCs and Bluetooth access points that connect mobile devices to the fixed systems. When mobile devices access and request a service from a fixed system, it creates an agent for managing them, which interact between agent and user.

The ANBD can provide agent service where TCP/IP is not supported, but must have a fixed infrastructure like JADE-LEAP. In addition, the ANBD creates the user agent on fixed infrastructure; this can limit the kind of agent services available.

3 Proposed system

In this paper, we propose the following:

- Provide agent service among mobile devices without a fixed infrastructure.
- Choose the ADSP and change it dynamically.
- Maintain the ADSP through another device when the current one disconnects from the MANET.
- Support agent communications between unreachable nodes.

As we described in Section 1 and 2, existing agent systems have the limitation that they can only be used on a fixed infrastructure (base stations of ad-hoc network, access points, etc). Using the above features, our platform is able to run without a fixed infrastructure, and provides white and yellow pages services in MANETs. Finally, we prevent the service interruption problem, which can occur in the network and support communications for the unreachable nodes.

3.1 System Architecture

We design the architecture to be compatible with FIPA specification and add only a module named AD Manager(ADM). Each module of the platform can communicate those of others' via the Message Transportation System. The ADM, an added module, consists of Comparer, Connector, Transmission Manager, AMS and DF Controller, Router and Proxy, as shown in Figure 1. Each module is explained as follows.

(1) Comparator(CP) – CP knows about the device's(it can be either mobile or fixed) hardware performance, and compares it with other devices in the network. By this, ADSP decides the next ADSP. After the system is composed, the CP of the ADSP manages the priority order of the next ADSP by performance of the others'.

(2) Connector(CN) – When the device connects to the MANET, the CN finds the ADSP in the network and registers its own agents and services to the ADSP, performs disconnection to other nodes in advance, and recognizes that the other node is disconnected or the ADSP of the system has changed. In addition, when it communicates with the ADSP, it transfers a list of connected nodes.

(3) Transmission Manager(TM) – To handover the role of the ADSP, the TM performs backup of agents and service data to other devices or holds the received data internally. The TM runs only on the ADSP or backup devices.

(4) AMS and DF Controller(ADC) – If disconnection of the ADSP is recognized by the CN, and the device should be the next ADSP using the priority order, then, ADC registers agents and service information to its own AMS and DF, and executes the agents. It runs when a change in the ADSP is needed.

(5) Router(RT) – The RT is responsible for routing pathes between unreachable nodes and sending messages through the paths.

(6) Proxy(PR) – The PR takes charge of proxying agent services. The PR registers transmitted agents from the requesting device to the AMS and DF, executes them, generates results, and communicates with the requester.

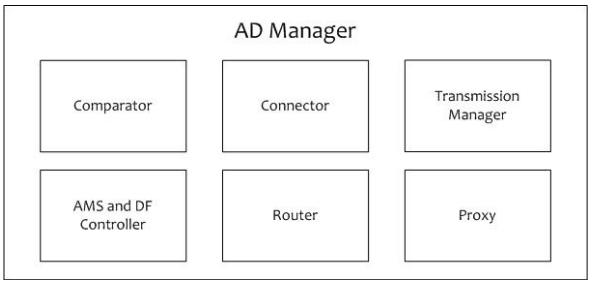

Fig. 1. Components of AD Manager

3.2 Dynamic composition of agent system

The ad-hoc network composed only by mobile devices may communicate in a Peer-to-Peer (P2P) manner. At this time, a node has to connect to other nodes in the network (composed by n nodes), and therefore it takes a time period of O(n). It is much more inefficient than when agent system is managed by the ADSP, which takes a time period of O(1) because a node can find the ADSP, by accessing the first node of the network. Therefore, the ADSP has to be the agent system in the MANET.

3.3 Choosing ADSP on MANET

There’s no problem of providing white and yellow page services in a fixed infrastructure system because computing power and network bandwidth are sufficient. However, an ad-hoc network that consists of mobile devices has limited ability to

support services. Although the amount of communication and computation in MANETs is much less than that in wired networks, in the view of hardware, the ADSP should be more powerful in the MANET. In this case, computing power is definitely the most important factor but battery power has to be considered. For this reason, when the initial network is composed by CN, CP of the more powerful device begins to provide services and after that, whenever new devices come into the network, the CP of the ADSP compare each device's hardware and changes the ADSP, or modifies the priority order of the next one list. At this time, the reliability and residence time of the next ADSP must be considered, to change the ADSP.

3.4 Maintenance of ADSP

If the ADSP is disconnected from the agent system, other devices on the network cannot be served white and yellow page services. Therefore, even though this occurs, it needs to maintain agents and services information and handovers the role of the ADSP to nodes in the network. It can be divided into two cases, where the ADSP breaks away from the system. Firstly, the ADSP exits the network and the user or agent notifies that it is going to exit. Secondly, the ADSP can be abnormally disconnected. For example, it can be from device errors or it could be out of communication range. The former transfers the data of the current AMS and DF before leaving the network. Of course, receiving data devices are decided by the priority order of CP on the ADSP. Thus, the former can deal with this simple method, but the latter cannot be handled in this way. For this reason, we propose to backup the data to other devices periodically. In other words, if more than four nodes compose the network, the TM of the ADSP would backup data to one of the other nodes cyclically. When the ADSP is disconnected irregularly, the backup node's ADC allows the system to provide services normally by taking the role of the ADSP.

The CN of the new ADSP notifies the change of the ADSP to all nodes' CN and resumes the service in both cases. The ADSP only provides white and yellow pages services, when the ADSP is changed, it does not affect the other nodes that are communicating between them already.

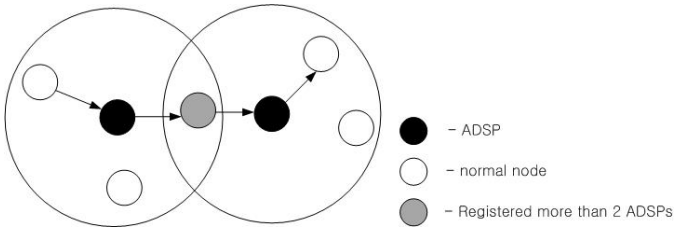

Fig. 2. Routing service to unreachable node

3.5 Secured routing service to unreachable nodes

The ad-hoc network has a limited range to communicate, one node may not be able to be served even though there are nodes that provide that service.[5] To solve this problem, our platform supports routing between nodes that want to communicate. It sets up paths composed by connected nodes to reach the target node, because the connection establishment takes longer in the MANET. It is done by each node's RT.

Figure 2 illustrates the routing service to unreachable node. If a node wants to get services from other nodes known by the AMS or DF, but the target devices are not on its device dectection list, it searches for routing paths to the ADSP of the system. At first, the ADSP sets up the routing paths using its node list, when routing fails, it searches on another ADSP. According to the location, a node can be registered by more than two ADSPs, and routing can be used in this inquiry. The ADSP obtains a list of connected nodes from each device, when it needs to communicate with them. Explict syncronization is not necessary for reducing the overhead of the ADSP. As the result, the ADSP does not guarantee success in routing because it sets up several paths based on non-recent information.

In this case, it is a problem that messages can be exposed to nodes of the paths. To solve it, we suggest applying a RSA public key cryptosystem. ADSP knows every node's public key and sends it when it provides white and yellow page services. The node, which starts to communicate, encrypts messages with the target node's public key and sends them with its public key, the target node decrypts messages with its own secret key. Then, the target node answers it by encrypting messages with the sender's public key. These steps can create overhead on mobile devices, and can be reduced by encrypting important messages and adjusting bits of keys.

4 Implementation and evaluation

As above, our platform is based on Bluetooth technology. For communication, Bluetooth uses service discovery after device detection. In this way, it can find whether the device runs the agent system or not. During the service discovery, it is possible to inform the ADSP of the network. It can solve the Bluetooth's problem that it has to connect to each device for service discovery. We implemented our system using Java and use IBM's J9 (KVM-JCL Personal Profile) [6] as the Java Virtual Machine and Avetana's library [7] as the Bluetooth stack. We evaluated it with three Bluetooth 2.0 devices and two Bluetooth 1.1 ones.

Figure 3 shows the time that the initial two devices take to compose the system. The first experiment took 56 seconds against the average of the others, which was 10 seconds, it was assumed that there was some interference because Bluetooth uses the same amount of bandwidth as the wireless LAN(802.11 standard).

After the system's composed, the time required for the new node to participate and register their agents and services to the ADSP was similar to that shown in Figure 3. This means that detection device time is much longer than registering the service time.

Table 1 represents the time from the other nodes recognizing unintentional disconnection of the ADSP to the next ADSP resuming services depending on the

priority. We measured this experiment by turning off the ADSP, the other devices recognized it, the next ADSP executed white and yellow pages services using backup data, and notified the others. If the next ADSP connects with the other nodes in the Piconet, the time spent changing the ADSP time will be short. However, we assumed the worst case that the next ADSP was not connected with any other nodes and tried to connect with all nodes.

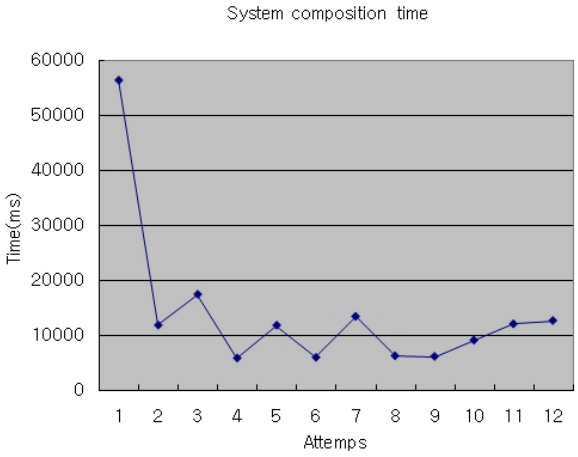

Fig. 3. Initializing time of the system

Table 1. Time to change ADSP (ms)

	(number of devices) 3	4	5
(attempts)1	3532	28281	50813
2	3781	29812	51922
3	4140	23719	56312
4	4125	29328	73688
5	3922	40093	68750

Although a system with 3 devices has a good transfer rate, the more devices that compose the network, the greater the transfer rate.

The time to route the message between nodes which are unreachable each other is in table 2. It was evaluated by the condition that the number of routing devices is 1, 2, and 3 with the exception of the ADSP's path setting time. The ADSP sets the path of messages from the sender to the receiver. It was assumed that there is no communication fault and the distance between communicating nodes are 5m approximately.

Each device can connect to the others more than one concurrently, so routing the message was done with short time. If the routing algorithm is optimized when there are many nodes in the network, it would be efficient.

Table 2. Time to route the message (ms)

	(number of routing devices) 1	2	3
(attempts)1	356	639	987
2	432	633	899
3	319	707	952
4	443	684	942
5	577	731	853

From these experiments, we found that the connection time is a significant determinate of the performance of the Bluetooth protocol. Bluetooth transmission rate is good after its connection, and we expect much more efficiency if device detection and connection time are shortened.

5 Conclusion

Mobile devices suffer from more limitations than PCs. The size of screen, input device, capacity of battery, computing power and unstable connection are all general factors that are limited in mobile applications. But there are much more important factors limited, such as composition, communication, and maintenance of network.

In this paper, we explain the need for the ADSP in MANETs based on Bluetooth, and it is suggested that this is changed dynamically to respond to the network environment. In addition, we propose algorithms for dealing with disconnection of the ADSP, provide encrypted routing between unreachable nodes and support agent services for devices that are unable to run the platform. Finally, we implement prototypes of the platform and demonstrate the efficiency of our algorithms.

Although hardware is constantly improving, mobile devices currently are limited in computing power. So, future work will focus on making a lighter-weight platform. In addition, we need to support other protocols to run on our platform.

References

1. IEEE Foundation for Intelligent Physical Agents (FIPA). Agent Management Specification. http://www.fipa.org/specs/fipa00023/SC00023K.pdf
2. JADE. Java Agent DEvelopment Framework. http://jade.tilab.com/
3. Java Agent DEvelopment Framework (JADE). JADE-LEAP User Guide. http://jade.tilab.com/doc/LEAPUserGuide.pdf
4. Alessandro Genco, Salvatore Sorce, Giuseppe Reina, Giuseppe Santoro, "An Agent-Based Service Network for Personal Mobile Devices," IEEE Pervasive Computing, vol. 05, no. 2, pp. 54-61, Apr-Jun, 2006.
5. Jamie Lawrence, “LEAP into Ad-Hoc Networks,” Workshop on. Ubiquitous Agents on embedded, wearable, and mobile devices, Bologna, 16th July, 2002.
6. IBM J9 KVM - Workplace Client Technology, Micro Edition http://www-306.ibm.com/software/wireless/wctme/
7. Avetana JSR-82 implementation. http://www.avetana-gmbh.de/avetana-gmbh/produkte/jsr82.eng.xml