EVALUATION OF RECENT RESEARCH IN MOBILE AGENT PLATFORMS FOR MOBILE DEVICES

Qusay H. Mahmoud, Zhujun Xu, Xining Li
Department of Computing & Information Science,University of Guelph, Guelph, Ontario, Canada N1G 2W1

Keywords: Mobile agents, mobile devices, wireless devices, security, scalability

Abstract: Most mobile devices such as cell phones suffer from limited computational resources as well as a small display size in which to present information to the user. In addition, the wireless environment that devices operate in suffers from limited bandwidth and frequent network disconnections. Mobile agents, which as software entities that are capable of roaming the network, do not need much bandwidth. This makes mobile agents a promising mechanism for the mobile devices. In this paper, we report on recent research that has been carried out in the area of mobile agents for mobile devices; we classify the several agent platforms that have been developed according to their underlining approaches and we discuss their advantages and disadvantages as well as limitations and concerns for embedded mobile agent platforms. In addition, we present directions for future direction in this area. To conclude, limitation and concerns for embedded mobile agent platforms are discussed.

1 INTRODUCTION

Mobile devices, such as cell phones, Personal Digital Assistants (PDAs) and pocket PCs have gained popularity for their distinguished advantages in today's information society. The capability of wireless access to Internet resources particularly indicates the potentially explosive growth of wireless applications on mobile devices. On the other hand, mobile devices have limited computational resources such as memory and processing power, low bandwidth, and a greater tendency for network errors. Such disadvantages inevitably have an impact on the response time and throughput of applications or services that mobile devices are accessing.

To meet with the demanding requirements of the users, wireless applications will have to be: (1) Self-adaptable in various environments which include the users, the mobile devices and other resources (e.g. network); (2) Interoperability with the users and other mobile devices; (3) Able of proactively providing and arranging rich personalized applications or services to the users; and (4) Capable of overcoming the low bandwidth and slow network connections;

Mobile agents represent a promising mechanism that can help with the development and deployment of wireless applications with the above capabilities. Mobile agents are autonomous pieces of software that act on behalf of their creators to proactively access the applications or services, or solve the problems for their users across the network. Mobile agents may have the capability of interacting (i.e. communicating, cooperating, negotiating) with other agents. Integrated with the artificial intelligence technology, agents may easily obtain the ability of learning from the changes of the environment or reasoning about the existing organizational structures. The biggest advantage of mobile agent technology that may benefit the wireless environment is the requirement of low bandwidth and support for disconnected operations. Mobile agents normally move to remote hosts where they can collect and process data locally without the risk of network disconnection. After completing their tasks, agents return to their home.

Given these advantages, finding how to deploy agent platforms on mobile devices is becoming a big challenge considering that not all mobile devices are capable of running an agent platform since they do not have enough processing power.

The rest of this paper is organized as follows: Section 2 introduces several approaches to the development and deployment of mobile agent

H. Mahmoud Q., Xu Z. and Li X. (2004).
EVALUATION OF RECENT RESEARCH IN MOBILE AGENT PLATFORMS FOR MOBILE DEVICES.
In *Proceedings of the Sixth International Conference on Enterprise Information Systems*, pages 331-336
DOI: 10.5220/0002642103310336

platforms for mobile devices, and discusses their architectural styles and implementations. The approaches are compared and contrasted in Section 3. Security concerns and requirements for protecting mobile devices from malicious agents are presented in Section 4. Section 5 discusses some scalability issues. Finally, concluding remarks are discussed in Section 6.

2 AGENT PLATFORMS FOR MOBILE DEVICES

A mobile agent platform is a software infrastructure that implements the mobile agent paradigm. The platform provides an environment supporting agents' creation, migration and execution. In the application level, the mobile agent platform should provide sufficient APIs for agent application developers or programmers to create and test agents running on it. From the user or agent point of view, a collection of services or applications should be provided in a mobile agent system.

The creation, execution, migration and deletion of agents are costly. This means that running mobile agent on mobile devices is very resource demanding. Currently, there are several approaches on the implementation of deploying agent platforms on mobile devices [Cosmin, 2003]. In order to run agents on resource limited mobile devices, most mobile agent platforms that runs on mobile devices must have a light-weight architecture which enables mobile agents to completely run on the devices; whereas, some platforms are placed on remote stationary server and mobile devices use remote mobile agents using a delegation model through wireless communication. A compromised approach is also proposed in [Wang, et al, 2003]. From the agent point of view, these approaches can be classified into three categories:

Remote Mobile Agent Platforms

In this kind of approach, agents are not executed on mobile devices. Instead, there is a light-weight client side application installed on mobile devices interact with a mediator running on the agent server located on a stationary computer. The client represents the user and communicates with mediator via is an interface for the agent platform. Through the client, users create and dispatch mobile agents running or migrating on remote agent servers. After accomplishing their tasks, states of the agents and data will be returned to the mobile devices by the client side application.

Customizable Mobile Agent Platforms

This approach compromises between the outside agent platform approach and embedded agent platform approach discussed above. The mobile devices are categorized based on evaluating the capability of the computing resources available. The device is configured with a thick agent client that is able to execute the mobile agent locally. Devices with limited resources use a thin agent client which only manipulates the agent's data.

Now, we discuss the basic characteristics of the three approaches as well as their advantages and disadvantages.

Embedded Mobile Agent Platforms

This approach makes the mobile agent platforms entirely deployed on the mobile devices. Due to the limited resource available, the mobile agent platform must have a light-weight architecture to provide limited services or applications. To meet the requirement, many techniques for downsizing the platform such as code profiling and romizing have been developed .

2.1 Remote Mobile Agent Platforms

This approach is based on the idea that mobile devices do not have enough processing power to run an agent platform locally. Users of such devices would be able access, use, and dispatch remote mobile agents running on wired networks. The MobiAgent system [Mahmoud, 2002] follows this approach.

The MobiAgent system architecture consists of four main components: the Clients, the Communication Manager, the Agent Gateway, and the MobiAgent Services. These components are connected by either wireless or wired links. The Communication Manager acts as a mediator between the mobile devices and the wired lines. It is responsible for the interaction between the mobile devices and other parts of the MobiAgent System. The Agent Gateway provides an interface to the Communication Manager and MobiAgent Services to create and update user profiles. When a user sends requests to the mobile agents though the Agent Gateway, the mobile agent will perform its task and go back to its host, passing the results to the Agent Gate after accomplishing the job.

In this approach, both mobile agents and platforms are placed on remote stationary servers. The only application running on the device is the Java 2 Micro Edition (J2ME) based Mobile Information Device Profile (MIDP) MIDlet [J2ME], which configures the mobile agent. Therefore, any mobile device supporting J2ME will be able to use this approach regardless of the power of the

computing resources available. The connection between the Clients and the Mediator is HTTP over a wireless link. The main advantages of this approach are:

- **A light-weight client in the mobile device**: By moving the very resource-demanding mobile agent platform to the resource-rich server, the major cost in the device is spent on interfaces.
- **Few security issues are required to be considered**: Since mobile agents are running on remote stationary servers, security issues such as protecting hosts are the responsibility of the security manager of the mobile agent system.
- **It overcomes low bandwidth and network disconnection**: Once the agent has been dispatched, the agent repository records the state changes and related results. When the user re-connects, she/he can freely check the results.
- **The system architecture is not tied to any agent platform**: In fact, any agent platform can be used in this approach.

The system architecture is suitable for resource-limited mobile devices, especially the less powerful cell phones. This approach, however, requires a stationary bridge between the mobile device and mobile agents.

2.2 Customizable Agent Platforms

The Mobile Agent Architecture for Heterogeneous Devices [Wang et al, 2003] is an example of this approach. The system architecture is composed of four main parts: the agent system repository, the general agent sever services, the mediator and the agent client. The agent system repository is the central part of the architecture which stores the information both for agents and their clients. The repository controls the services of agents such as registration, locating and management of the agent lifecycle. The system here evaluates the device capability and sets up the thick agent client for the device with powerful CPU and sufficient memory. The thick agent client can execute the mobile agent locally. While in the resource-limited device, the thin agent client is configured that is only able to manipulate the agent's data. The mediator consists of the agent initiator, the agent joiner/splitter and the mobile agent dispatcher. The agent initiator initiates the agent clients first time or reconfigures it for some changes on the resources of device or the software installed. The mobile agent dispatcher is responsible for dispatching agents between clients and servers. The agent joiner/splitter splits or joins the data and code of the mobile agents during their migration.

This approach provides a flexible mobile agent architecture which makes it possible for various devices to access the mobile agent systems. It is feasible to install different agent clients based on their respective capabilities. Also separation of the agent code and data enables either a full mobile agent or a light-weight version of the agent (data) running on the device.

A major disadvantage of this approach is that it does not support advanced interactions with the user or computations required at the mobile client. In addition, switching the client from thick to thin and vice-versa makes for an inconsistent experience for the user. The thin clients have to pay expensive bill for joining or splitting the agent code and data. But since all these resource-demanding computations are on the remote stationary hosts and the client is light-weight, this is not a big issue.

2.3 Embedded Mobile Agent Platforms

In the previous two approaches, the mobile agent platform is deployed on stationary host in a wired network. But as mobile devices become more and more powerful, they will have sufficient computing resources to run an entire agent platform. Two standardization efforts, namely FIPA [FIPA] and OMG/MASIF which aim to help standard agent platforms to fit mobile devices, are underway. In this section we describe three agent platforms that are meant to be embedded on the device.

2.3.1 Tryllian's Agent Development Kit

Tryllian's Agent Development Kit (ADK) [Tryllian] is a complete development and implementation environment. The ADK has two important components: the Agent Foundation Classes (AFC) and the Agent Runtime Environment (ARE). Applications build the agents on the basis of AFC and use the ARE to run them. Agent applications typically consist of several Habitats spread across a number of servers, each hosting several Rooms. A Habitat is a collection of Rooms that share a Java Virtual Machine (JVM). It provides services such as agent lifecycle management, communication, inter-Habitat travel, Room and agent persistence and the basic security model. Agents can only exist in Rooms. Rooms are the living environments where agents interact. Agent communication is done through messaging. The ADK provides a real-time, peer to peer architecture with dynamic update, load balancing and fault tolerance.

The significant features of this approach are the support of strong mobility and the full security model. Agents can move to other Habitats during

runtime, taking along with their (new) Java code and state. The ADK implements the full Java security model using certificates and permissions. The system is well protected against malicious local or foreign agents by strict control of agent permissions. The ARE protects agents from each other. Additionally, all the transmissions are under protection of encryption by using a secure socket layer (SSL). More powerful features provided in ADK means more computing resource required. At present only powerful mobile devices such as PocketPCs can support the execution of the ADK. The ADK is FIPA compliant and supports both J2SE and J2ME platforms. The Tryllian's Agent Development Kit is commercially available.

2.3.2 MicroFIPA-OS

FIPA-OS [FIPA-OS] is a component-based toolkit enabling rapid development of FIPA compliant agents. MicroFIPA-OS [MicroFIPA-OS] is an agent development toolkit and platform based on the FIPA-OS agent toolkit. The target devices of MicroFIPA-OS are non-resource-constrained devices. To fit on resource-constrained mobile devices, MicroFIPA-OS simplifies the FIPA-OS components. XML parsing and heavy use of threads are removed. Thread and resource pools are employed. In addition, MicroFIPA-OS limits the use of the Java reflection API and dynamic invocations.

The MicroFIPA-OS basically follows the FIPA-OS architecture and it's extensible by plugging in components that either replace or supplement the architecture [FIPA-OS]. FIPA-OS programming API is also supported in MicroFIPA-OS with some limitations pertaining to the supported interaction protocols. The platform can be run in a minimal mode, which enables the agent to be smaller by using some low-level programming API. Programmers or developers may create their own task and conversation mangers in this mode, and better performance can be achieved.

A major limitation of MicroFIPA-OS is its limited support for mobile devices. At present it is only suitable for some high-end PDAs.

2.3.3 LEAP

The Lightweight Extensible Agent Platform (LEAP) [Bergenti et al, 2001] is the most well-developed agent platform for mobile devices. LEAP is based on JADE [Bellifemine et al, 1999], which is an agent platform developed under a grant from the European Commission. The JADE framework, which is fully implemented in Java consists of a main container and several agent containers. In an agent container, there is a message dispatcher responsible for delivering messages of agents to other containers and an Agent Communication Channel (ACC) responsible for passing messages between different agent platforms. A main container is a special case of an agent platform. The main container maintains special services on the platform such as the Agent Management System (AMS), the Agent Communication Channel (ACC) and the Directory Facilitator (DF) serving as the white and yellow page directories where agents can request each other's address and check whether agents exists anywhere that perform special services.

LEAP is the result of an EU project of a consortium consisting of Motorola, BT, Siemens, and others. The LEAP, combined with JADE, replaces some parts of the JADE kernel forming a modified runtime environment that is identified as JADE/LEAP ("JADE powered by LEAP"). The JADE/LEAP can be deployed on a wide range of devices varying from servers to Java enabled cell phones.

We believe that embedding agent platforms in mobile devices is the most promising research direction. The tight integration approach should yield greater flexibility and be most effective way to design and control agents. However, there's a major limitation in this approach: The low level of computing resources available, which may directly limit the agents' competencies. Mobile agent systems today are very resource demanding both for the client and the server. As a result, though mobile devices are experiencing a continuous growth in resources and power, some of them still do not have the capability to execute a mobile agent platform. For example, the JADE/LEAP platform supports agent mobility between J2SE and Personal Java containers, while mobility is not available on MIDP. In other words, mobile agent technology would not be applied in the great majority of Java enabled cell phones on which JADE/LEAP is installed.

3 COMPARISON

This section presents a comparison between the three different approaches discussed above, namely "remote", "customizable", and "embedded". The comparison is based on the following issues:

- **Mobile agents**. In the "remote" approach, users run MIDlets on their mobile devices to dispatch a remote mobile agent using a delegation model. All agents execute and migrate on stationary agent servers. The "customizable" approach enables the entire agents (code and data) run in Thick Clients, but in the case of Thin Clients,

only part of the agent (data) can be accessed. When the thin agent (data) returns to the stationary server from the mobile device, two parts will be joined and become a full mobile agent. In the "embedded" approach, the lifecycle of the agent including creation, execution, migration and deletion, is completed on the mobile device. It's worth noting that supporting agent migration is very resource-demanding; therefore, in the latter two approaches, the agent mobility is weak in most cases.

- **Security issues**. Mobile agent security is an open research area. Security issues include securing communication, protecting hosts against malicious agents and protecting agents against malicious hosts. Implementing a full security model is very expensive, especially for the deployment of the mobile agent system on mobile devices. The major advantage of the "remote" approach is that it moves the security requirements, except for protecting network data, to the security manager of the stationary mobile agent platform. And securing the communication via HTTP is relatively easy and less costly. For the "customizable" and "embedded" approaches, a minimal security model is supported.

- **Targeted devices**. The "remote" approach enables any mobile device to access and use remote mobile agents as long as it supports J2ME. In the "customizable" approach, thick and thin clients are available to a variety of mobile devices. In the "embedded" approach, however, the more features are supported the less mobile devices are able to run the platform.

- **Implementation standard**. The mobile agent platform located on the remote stationary server cam be FIPA or MASIF compliant, but not mandatory. Most embedded mobile agent platforms on mobile devices are FIPA or MASIF compliant and some support both standards. For example, the Tryllian's Agent Development Kit and JADE/LEAP are mainly FIPA compliant.

- **Usability**. Usability evaluation mainly relies on the interaction methods between a user and a wireless application such as screen resolution, input mechanism and user interaction patterns. Compared to "remote" and "embedded", the "customizable" approach is not satisfactory as it doesn't support advanced interactions with the user or computations required at the mobile client. In addition, switching the client from thick to thin and vice-versa make the operation inconsistent.

- **Development**. Since the "remote" approach may use any mobile agent platform available, the key points of development focus on the non-agent-platform components such as Communication Manager and the Gateway, which provide the interface between the users and the MobiAgent services. For the "embedded" approach, it's very important to make the platform light-weight and suitable to run on a variety of mobile devices including cell phones. The "customizable" approach is complex because of the configuration of the clients, the implementation of the agent joiner/splitter and its interaction with the agent server all lead to heavy development.

4 SECURITY

The integration of mobile agents into mobile devices will not be effectives and useful unless effective security mechanisms for both, mobile agents and mobile devices are adopted. For instance, current techniques available are not able to determine whether a foreign agent is malicious or not, unless the important data on the mobile devices are damaged. Security issues in a mobile agent system can be classified into two main categories: (1) Protecting agents against malicious platforms; and (2) Protecting mobile devices and the network against malicious agents. Protecting mobile agents against each other is another problem, which might be ignored if the agents are cooperative agents and can trust each other.

The malicious host problem, in which a malicious host attacks a visiting mobile agent is the most difficult one. In order for an agent to function, the host on which it executes must be able to interact with it. Most mobile agent systems rely at least in part on virtual machines to standardize the execution environment. However, agents written in scripting and interpreted programming languages are easily de-compiled from bytecodes to their original sources. Therefore, a malicious host may easily steal private information, modify the agent code or mislead the agent. At present the security for protecting mobile devices against malicious agents is a major concern.

In the remainder of this section we will discuss the security requirements for protecting mobile devices against potential malicious agents and offer possible solutions.

Migration from un-trusted hosts

Mobile agents from un-trusted hosts may carry valuable or useless information. Existing security

mechanisms assume that trusted hosts wouldn't dispatch mobile agents carrying useless information. To reduce the potential attacks, the security manager in the mobile agent platform must be able to accept or reject migration request from un-trusted hosts. Special mechanisms also need to be provided to monitor and fully control the activities of foreign agents from un-trust hosts.

Authentication

Many mobile agent systems on stationary servers use user information/profile registry in an external directory for authentication. However, it's very resource demanding and requires a reliable communication channel, thus not suitable for mobile devices. To be efficient and safe, the authentication should be performed locally.

Access control

Some platforms employ strict access controls while some prefer the flexible access control. In the former approach all the foreign agents are considered potentially malicious and very strict access policy is applied. It reduces the potential attacks from malicious agents but offers less flexibility. Mobile agents carry on various tasks and they may require different access permissions on resources to accomplish their jobs. For the sake of extensibility and flexibility, a customizable access control mechanism is a better choice for platforms on mobile devices.

Interaction

In order to protect the local resources, providing some monitoring toolkit will help users deny illegal or potential harmful behaviors of agents. Since the mobile devices have limited resources and usability constraints, simple user interfaces will help improve the interaction.

5 SCALABILITY

As the number of agent and their activities increases so does the number of threads. Such increase leads to more consumption of system resources and may be the instability of the platform. This may affect the system performance. To resolve the scalability problem, several issues need to be considered [Cosmin et al, 2003], including the suspending of inactive agents.

When an agent is in an inactive state, it should be put in a repository and removed from memory. In other words, the agent is still present and not consuming resources such as memory and processing power. If communication is needed with such inactive agent, it will be automatically deactivated.

6 CONCLUSION

In this paper we discussed, classified, and evaluated three main approaches for developing and deploying mobile agent platforms on mobile devices. Each approach has its own advantages and disadvantages, as well as domain-specific applications. It is hard to say which approach is the most suitable for deploying mobile agents on mobile devices given that mobile devices continue to have varying capabilities such as processing power, memory, and display size. While the "embedded" approach sounds the most ideal and promising, several security issues need to be researched further.

REFERENCES

Bellifemine, F., Poggi, A., Rimassa, G., 1999. JADE – A FIPA-Compliant Agent Framework. In PAAM'99, Conference on Practical Applications of Intelligent Agents and Multi-Agents, London, UK.

Bergenti, F., Poggi, A., 2001. LEAP: A FIPA Platform for Handheld and Mobile Devices. In ATAL'01, 18th International Workshop on Agent Theories, Architectures, and Languages, Seattle, USA.

Cosmin, C., Boissier, O., 2003. Multi-Agent Platforms on Smart Devices: Dream of Reality? In SOC'2003, Smart Objects Conference, Grenoble, France.

FIPA:
http://www.fipa.org

FIPA-OS:
http://fipa-os.sourceforge.net

J2ME:
http://java.sun.com/j2me

Mahmoud, Q.H., 2002. An Agent-based Approach to the Wireless Internet. In Journal of Internet Technology, Vol. 3, No. 2.

MicroFIPA-OS:
http://www.cs.helsinki.fi/group/crumpet/mfos

Tryllian Agent Development Kit:
http://www.tryllian.com

Wang, A.I., 2003. A Mobile Agent Architecture for Heterogeneous Devices. In WOC'03, 3rd IASTED International Conference on Wireless and optical Communications, Banff, Canada.