Service Delivery Agent System for Mobile Devices

Jeong Seob Yoon, Lee Ki Hyun, Geun Sik Jo

Department of Computer Science and Engineering, Inha University

253 Yonghyundong, Namgu, Inchon, 402-751, Korea

Tel: +82-32-875-5863, Fax: +82-32-875-5863, E-mail:jsyoon@eslab.inha.ac.kr

Abstract

Recently the wireless-internet has been spreading extensively. People are spending a large part of their time gaining access to information using a mobile device. With the rapid growth of on-line Electronic Commerce, the use of mobile devices creates a new paradigm that provides users with location-independent real time service. Although this new paradigm does have some advantages, limited process speed, low bandwidth, the low battery capacity of mobile devices, and a high rate of wireless network errors causes many overhead expenses during service time with the server. In this paper, we suggest an autonomous service delivery system, which provides mobile agent capability to users that cannot maintain a connection. We have developed the system based on java mobile agent technology. Using this system, we can provide more effective service to users when the user is sending requirements for service through a mobile device that has limited resources. Furthermore we can manage the contact server dynamically when new services are added.

Keywords : mobile agent, mobile device, java

1. Introduction

As the wireless Internet is growing rapidly, the rate of access with a mobile device is also rising. Mobile users are encountering the unique characteristics of mobile access: timeliness of data provision and accessibility away from the home or office. These characteristics release mobile users from the time and location of access information. However, when mobile users maintain a long-time connection with the service server, the disadvantages of the mobile device, such as low bandwidth, limited process speed, poor battery capacity, and frequent connection error with the wireless network, cause many problems. Also, in the current mobile environment, if services are provided using a CGI program based on HTTP, users must maintain an exclusive channel and services which are not enough to reflect the user's complex strategy because of the simple parameter passing method.

Mobile agent technology can offer a solution to help solve these problems. By using the mobile agent, mobile users need not maintain a connection between the mobile device and the service server. The mobile agent moves to the remote host with the user's instructions and can execute asynchronously and autonomously.

In this paper, we suggest an autonomous service delivery system called MAWS(Mobile Agent Based Wireless Service) in the wireless environment. Our system uses mobile agent technology in order to overcome the limitations of a wireless network. This paper is organized in the following manner: In Section 2 we will describe related works : the mobile agent, J2ME, and SMS. Section 3 presents our system architecture. Section 4 describes the service management. In Section 5, we describe mediating agent. Section 6 shows how to implement applications, and finally in Section 7 we discuss concluding remarks.

2. Related work

2.1 Mobile Agents

Mobile agents are intelligent agents capable of moving over a network, interacting with foreign hosts, gathering information on behalf of the user, and returning to the user after performing their assigned duties. Many groups have developed mobile-agent systems and agent software, for example, Aglet[1] , Voyager[2], Concordia[3]. To implement our system, we have selected the IBM Aglet System Development Kit(ASDK). Aglet Technology is a framework for programming mobile network agents in java developed by the IBM Japan research group. The ASDK provides Tahiti, which is an Aglet server and can be set up in each site to offer an execution environment for the coming Aglets [1].

2.2 J2ME(Java 2 Micro Edition)

Java 2 Micro Edition is a cross platform programming

language that can be embedded into small application environments, such as mobile phones and the PDA, or into less conventional devices, such as onboard car computers and household heating systems. The J2ME environment can be altered to suit the device through the Connected Limited Device Configuration (CLDC). This specifies if the device is battery operated, memory constrained, and processor limited, and if it has a low bandwidth and high latency network connectivity. The Mobile Information Device Profile (MIDP) is the set of Java APIs (Application Programming Interface) that provides the runtime environment for J2ME applications. The MIDP manages the application, the user interface, the networking and I/O [4][7].

2.3 SMS

Short Message Service (SMS) is the ability to send and receive text messages to and from mobile devices(mobile phones). Each short message can be up to 160 characters is length when Latin alphabets are used, and 70 characters in length when non-Latin alphabets such as Hangul (Korean language) and Chinese are used. Furthermore SMS is a store and forward service. In other words, short messages are not sent directly from sender to recipient, but always via an SMS center [5].

3. MAWS System Architecture

The architecture of the MAWS system is illustrated in Figure 1. The MAWS system consists of four components: the Mobile Device, the Mediating server, the Service Server, and the SMS Center. The mobile device and the mediating server are connected by wireless links, and the mediating server and service servers are connected by a wire-line. The mediating server is a connection point between the wireless network and the wire network.

In our system, the mediating server provides mobile users with various agent interfaces. The mobile user can generate the service agent indirectly through the agent interface. Also, the service provider can add new service through the mediating server. The mediating server adds new service agents to internal storage and manages the service list.

Marked arrows in Figure 1 represent the sequence of the system service mechanism.

(1) The mobile user downloads the agent interface and disconnects from the mediating server. Then the mobile user sets parameter values through the agent interface.

(2) The mobile user sends the service request message to the mediating agent.

(3) The mediating agent activates the service agent.

(4) The service agent goes to the service server.

(5),(6) The service agent performs necessary transactions to fulfill the request and returns the result to the mobile user.

MAWS allows the mobile agent to travel to the service server and send results to the mobile user after performing its assigned duties, even when the user is disconnected from the wireless network. This allows mobile users to overcome the limitations of the mobile device when mobile users want to receive wireless information service.

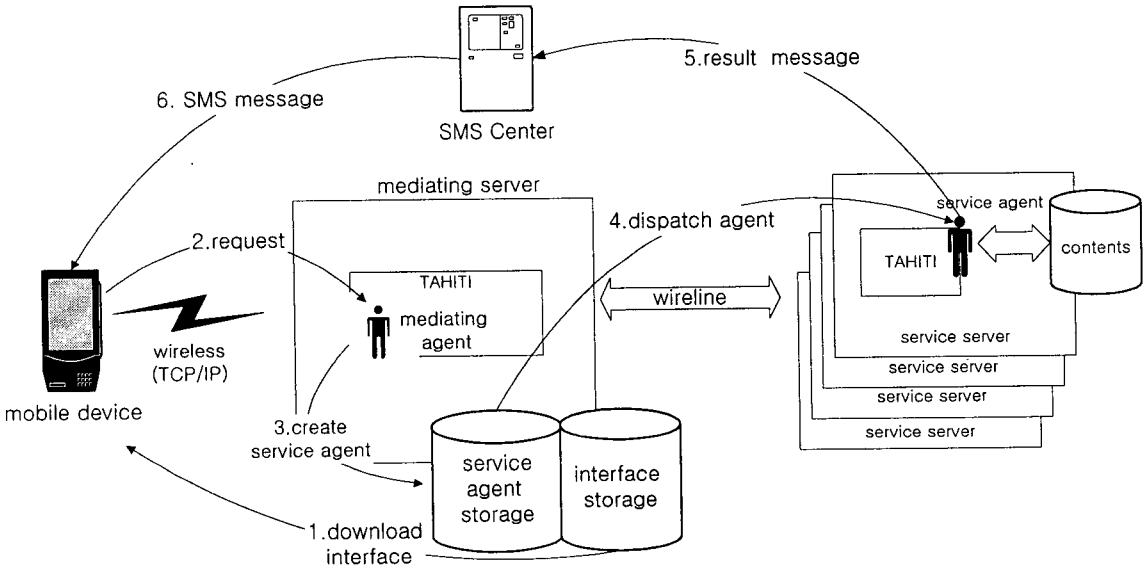

Figure 1: The main components of the MAWS system

4. Service and Interface Management

Service servers are Internet hosts that provide mobile users to various information services. Service servers have their own service agent that performs a task in their own service area: (i.e. Book search agent for online Bookstore, CD search agent for music shopping mall). Also, each service agent has its own agent interfaces. With respect to service providers, by separating agent interface for a service agent, they are a benefit to users who want to download files and keep a small program in limited mobile device environments. The agent interface is the input menu that is needed for a mobile user to customize the service agent. Each service agent and agent interface is implemented by a service provider and is stored in the mediating server.

In our System, service providers implement mobile agents (service agents) instead of mobile users. Therefore, mobile users do not need to acquire the skill to program the mobile agent (service agent). Instead, mobile users download an agent interface to customize service agents. This was done for the following reasons:

- Although some users may design service agents by themselves, most users do not have the ability to do so [6].
- It is more convenient if the service agent can be customized according to its own specifications by using the agent interface [6].
- A malicious user can create a service agent that will attack a service server. However a service agent implemented by the service provider is honest.
- A Downloaded interface can be reused to create the service agent. The mobile user need not download the same agent interface again from the mediating server.
- Whenever the service providers want, they can add new services to the system by implementing a new service agent and a new agent interface.

The implemented service agent is assigned a unique ID number, and then stored in agent storage. In addition, the agent interface is assigned an ID number the same as service agent's ID number and is stored in interface storage.

When the mobile users connect to the mediating server to download the agent interface, these users are able to select a desired agent interface. The selected agent interface is downloaded to the mobile device. The agent interface may be composed of several input fields. For example, the agent interface used to search for a CD may be composed of 4 fields: CD Title, Artist, Label, and Song Title. Then, the mobile user enters the parameter values through the agent interface in order to customize the service agent. Finally a service request message is sent to the mediating agent.

5. The Mediating Agent

The Mediating agent 's primary function is to determine what the user wants from a service request and activate a service agent in place of the mobile user. The service request message from the mobile user transfers to the mediating agent together with its customized data. The service request message is a datagram that is composed of an interface ID, a device ID (phone number), and parameter values. Figure 2 show the structural elements of the service request message.

Fields	Description
Interface ID	ID number that is assigned to the agent interface. This is needed for the mediating agent to search for a service agent from storage.
Device ID	ID number that is assigned to a mobile device (i.e. phone number). This is needed for the service agent to send the result to the mobile device, via the SMS center [8].
Parameter values	Agent interface is an input menu that is composed of several input fields. This is a set of values that is entered by the mobile user through the Agent interface.

Figure 2: The service request message field

The mobile user can enter the various parameter values through an agent interface. For example, if an agent interface is formed to search music, the mobile user will enter the musician, the song title, and so on. Then, parameter values, the interface ID and the device ID are packed into the datagram. The mobile user then sends the datagram (service request message) to the mediating agent.

When the mediating agent receives the service request message, he extracts the Interface ID from the service request message and selects the service agent with the ID number that is the same as the Interface ID from the agent storage. Then, the mediating agent customizes a selected service agent according to the parameters and the device ID.

The service agent goes to the service server and performs the necessary transactions to fulfill the request (parameter values). When the work is finished, the service agent returns the results to the SMS center. Then, the SMS center identifies the mobile device by the device ID, and the SMS center sends the results to the mobile device,

6. Implementation

In our experiments, we constructed the mediating server and simple service servers: a CD shop and bookstore. We only stored simple contents for the processing of a transaction with these two service servers. We set up the tahiti (an execution environment for the Aglets) in the mediating server and each service server. Then, we implemented the agent interface and the service agent for each of the service servers. We used J2ME (MIDP) in order to implement the agent interface and ASDK in order to implement the

service agents and the mediating agent. Figure 3 shows the tahiti and the mediating agent that exist in the mediating server.

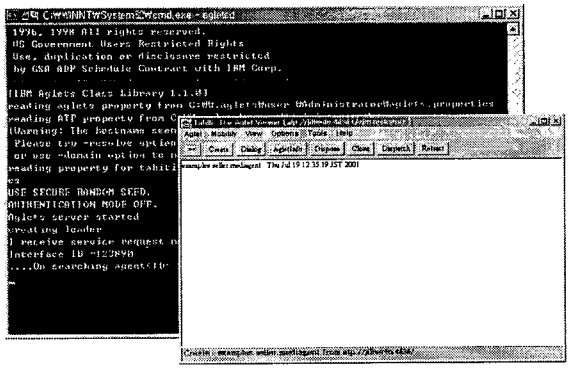

Figure3: The tahiti and the mediating agent in the mediating server.

In addition, we used a phone emulator [9] instead of a real mobile phone for experiments. The phone emulator is software that emulates the hardware of various models of mobile phones. This is extremely valuable for writing, testing and debugging applications. Lastly we implemented the SMS gateway program to simulate the SMS center. We stored each service agent and agent interface with the mediating server. Of course we were able to add further service servers to the mediating server. When the mobile user connected to the mediating server, the mobile user selected the agent interface — the agent interface for the CD search or the agent interface for the book search. The mobile user downloaded the agent interface, and he (she) disconnected from the mediating server. Figure 4 show the agent interface that is downloaded to the phone emulator. The user downloaded an agent interface for a CD search. After the user entered parameter values, such as "Song Title", "Artist", "Label", the device sent a service request message to the mediating agent. The mediating agent then activated the service agent with the ID number that was the same as the interface ID. After the service agent completed the task, it returned the results to the phone emulator via an SMS gateway(See Figure 5).

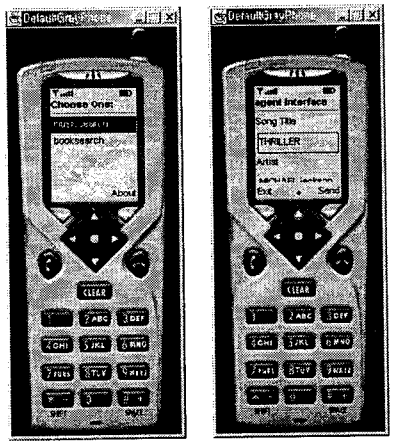

Figure 4: Agent interface.

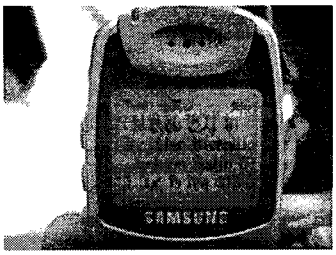

Figure 5: The result (SMS message)

7. Conclusion

In this paper, we have proposed the MAWS system as a way to eliminate the poor characteristics of mobile devices and the wireless network: low bandwidth and a high rate of wireless network errors. These poor characteristics have caused many problems when mobile users receive wireless information service. In our system, we have used mobile agent technology to resolve these problems. The mobile user need not maintain the connection to the service server for a long time using the mobile agent. The mobile user can indirectly activate the mobile agent by using the agent interface. The mobile agent can then perform the necessary transactions to fulfill the request, and return the results to the mobile user.

Acknowledgments

This work was supported by a grant No. 2000-1-30300-016-3 from Korea Science & Engineering Foundation.

References

[1] Danny B. Lange, Mitsuru Oshima, "Programming And Deploying Java Mobile Agents With Aglets", Addison-Wesley , pp.1-16, 1998.

[2] Voyager: http://www.objectspace.com/products/voyager

[3] Concordia: http://www.opencommunity.com

[4] Eric Giguere ,"Java 2 Micro Edition", Wiley, pp. 89-126, 2000.

[5] SMS: http://www.gsmworld.com/technology/sms.html

[6] Fangming Zhu, Sheng-Uei Guan, " SAFER E-Commerce: Secure Agent Fabrication, Evolution, & Roaming for E-Commerce", Idea Group Publishing

[7] J2ME: http://members.aol.com/javamobiles/

[8] SMS: http://www.yesm.co.kr/index.jsp

[9] Emulator: http://www.mobilejava.co.kr/bbs/main/board.jsp?code=midpboard