

Using JADE-LEAP to implement agents in mobile devices

Antonio Moreno, Aïda Valls, Alexandre Viejo

Grup de Sistemes Multi-Agent (GruSMA)
Departament d'Enginyeria Informàtica i Matemàtiques
Escola Tècnica Superior d'Enginyeria (ETSE)
Universitat Rovira i Virgili (URV)
amoreno@etse.urv.es, avalls@etse.urv.es

Abstract

Having frameworks that allow the implementation of personal agents on mobile devices, such as JADE-LEAP, is essential towards a future massive deployment of service-providing agents and their widespread social acceptance. In this paper we show how JADE-LEAP may be used to implement a personal agent on a PDA. This agent belongs to a multi-agent system that allows the user to request a taxi in a city. The personal agent communicates wirelessly, via Bluetooth, with the rest of the agents of the multi-agent system, in order to find the most appropriate taxi to serve the user.

1 Introduction

In the knowledge era we are living, citizens keep demanding fast, reliable, cheap, friendly and flexible ways of accessing any kind of information. Moreover, they want to be able to get this information from anywhere, anytime, using the latest technologies developed in the *Information and Communication Technologies* field, such as mobile phones, portable PCs or PDAs (*Personal Digital Assistants*). The current trend seems to go in the direction of joining these tools into single pieces of hardware with multiple capabilities.

It has been argued in the last years that *intelligent software agents* [1] have a collection of properties that make them very adequate to provide services to citizens [2]. Among these features we can remark the following:

- *Autonomy*: they can perform their tasks without a direct and continuous guidance from the user.
- *Learning*: they can apply machine learning techniques to construct automatically a user profile and adapt their actions to the user's preferences.

- *Proactiveness*: a personal agent can anticipate the needs of the user and perform tasks that may be beneficial for him, without an explicit request from the user.
- *Social ability*: a personal agent can get in touch with other agents that provide information about any domain in which the user may be interested (restaurants, cinemas, tourist events, medical centres, etc.), request the information that the user needs and present it in a friendly and personalised way.

The key point towards a widespread acceptance and use of personal agents by citizens is its deployment in the mobile devices that they normally use: PDAs and mobile phones. Therefore, it is of essential importance to build frameworks that allow developers to implement these personal agents in such mobile devices. Moreover, if personal agents have to be able to communicate with other service providing agents, they must follow a set of standard norms concerning the agent communication language to be used, the communication protocols to be followed, etc. There is not a universally accepted set of standards for developing multi-agent systems; however, the suggestions of FIPA (*Foundation for Physical Intelligent Agents*, [3]) are gaining a growing acceptance and are beginning to become a *de facto* standard in this field. Therefore, having frameworks such as JADE-LEAP (*Java Agent Development Environment-Lightweight Extensible Agent Platform*, [4]), that allow to develop FIPA-compliant multi-agent systems in mobile devices, is essential for the social acceptance of this technology and its take-up by citizens as a usual tool in their daily lives.

In this paper we report the experience of having successfully used the last version of the JADE-LEAP environment to implement a personal agent in a PDA. In the case study presented in this paper, this agent provides the user with a graphical interface that may

be used to request the services of a taxi. The personal agent gets in touch wirelessly with other agents that are running in a fixed PC to find out which is the most adequate taxi to perform the service, and shows this information to the user through the graphical interface. The structure of this paper is the following. First, we describe the basic characteristics of the JADE-LEAP environment and how its features can be used to develop personal agents on mobile devices. After that, we explain in detail how we established a wireless connection between the agents of the system. The description is illustrated with an specific example that has already been fully developed. The paper finishes with a conclusion, a glossary with the basic terms and the bibliographical references.

2 JADE-LEAP on PDAs

2.1 Overview of JADE-LEAP

JADE is a set of Java classes that allow a developer to build a FIPA-compliant multi-agent system quite easily. It provides a set of graphical tools that facilitate the complex task of implementing a multi-agent system. A few months ago an add-on to JADE, called LEAP, was released. This add-on replaces some parts of the JADE kernel, creating a modified environment called JADE-LEAP ([4]), which allows the implementation of agents in mobile devices with limited resources. It provides three modes of work to adapt to different circumstances:

- j2se: it can run in PCs with jdk1.2 or superior.
- Pjava: it can run in devices such as PDAs that have PersonalJava.
- MIDP (Micro Edition): it can run in devices that support MIDP1.0, such as mobile phones.

The three versions offer the same API to developers (there are minor differences from midp to the other two, though). Fig. 1 shows an example of the JADE-LEAP execution environment. In this example there is a multi-agent system with six agents. Two of them are running on the main container of the platform, which is connected through Internet with a container on a PC that holds another two agents. These four agents can communicate wirelessly with two agents running in mobile devices (one on a PDA and one on a mobile phone). Note that there is one container in each mobile device.

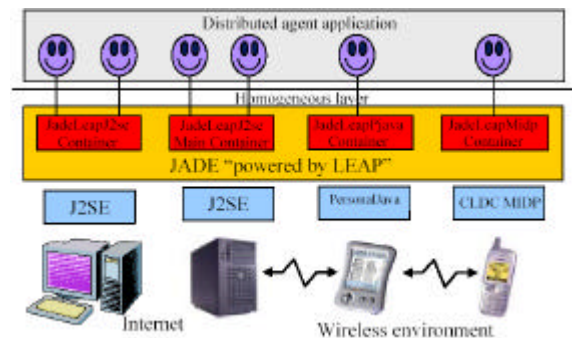


Figure 1. JADE-LEAP execution environment [5]

2.2 Wireless connectivity

The personal agents that are running in mobile devices have to be able to communicate (wirelessly) with service-providing agents running in fixed PCs. JADE-LEAP works at a very high communication level, establishing TCP/IP connections between containers, without caring about the physical means by which these connections are actually performed. Therefore, the application developer has to find a way of providing the connection. Two basic decisions have to be taken: the communication protocol and the connection type.

In our application we aimed to implement personal agents on *Fujitsu-Siemens Loox 600* PDAs. These devices have a WindowsCE 3.0 operating system, that supports *point-to-point protocol* (PPP), member of the TCP/IP set of protocols. Therefore, we decided to try and establish a PPP between the PDA and a fixed station.

Concerning the wireless connection, there is a wide range of alternatives; the most popular are GPRS, UMTS, WLAN, Wi-Fi, Bluetooth and infrared. As the aim of our project was to study the feasibility of implementing agents on mobile devices using JADE-LEAP, and not to make a commercial application, we chose the cheapest alternative: Bluetooth (which is supported directly by the Fujitsu-Siemens Loox 600 devices). The fixed PC did not have built-in support for Bluetooth, so we added a Conceptronic Bluetooth Dongle for this purpose. The basic shortcoming of this alternative is the connection range (around 10 metres). However, it suited perfectly our need of having an inexpensive wireless connection for our academic exercise. As stated above, the code of the agents does not depend on the connection type, so it could be used with any other type of physical connection without any modification.

2.3 Establishing the connection

It turned out to be quite complex to establish the Bluetooth connection between the PDA and the PC (which runs under WindowsXP). We were finally successful using the Microsoft ActiveSync 3.6 application. It was necessary to follow all these steps to get the connection up and running:

- Install Plugfree 2.0 in the PC [6].
Previously you have to uninstall all the software and drivers from the Conceptronics Bluetooth Dongle (in this way both the PDA and the PC use Plugfree to manage the PPP connection).
- Make the first connection between the devices.
The devices find each other and store the Bluetooth address of each other. We also define access passwords in this step.
- Install NetBEUI in the PC.
In the WindowsXP CD we look for the folder Valueadd\MSFT\Net\NetBEUI and copy the file Nbf.sys to C:\Windows\System32\Drivers and the file Netnbf.inf to C:\Windows\Inf. Then we add NetBEUI to the network configurations used by Bluetooth.
- Install Pockethost in the PDA.
It can be downloaded from the web page <http://www.handango.com/PlatformProductDetail.jsp?productId=54305>.
- Set up appropriate IP address and subnetmask in the Bluetooth Dongle network configuration.
- Set up appropriate IP address, subnetmask, default gateway and nameserver in the PDA (choosing Rappore technologies network adapter). Select the pockethost on the PDA (the computer name has to match exactly the one selected on the panel options of the PDA's ActiveSync).
- Set up the ActiveSyncs in the PC and the PDA.
First, we establish a partnership between the devices using the serial cable. On the connection settings of the PC's ActiveSync, select Allow USB and Allow Network and RAS, but do not select the option Allow Serial Cable. On the PDA's ActiveSync, we select the name of the partnership in the "Include PC when syncro" label. In "Enable Syncro when cradled" we select the Local Area Network associated to the PC Bluetooth.
- Connect the devices
Run Plugfree on the PDA and the PC. Establish a Personal Network connection from the PC. Having made that, Active Sync starts automatically both on the PC and the PDA. From this moment on, both devices are connected using PPP via Bluetooth, as

desired. Then we can start the JADE-LEAP platform on the PC and the PDA.

3 A case study

3.1 Multi-agent system architecture

We have chosen a specific application to test the implementation of personal agents in mobile devices. We have implemented a multi-agent system that allows to manage efficiently the set of taxis available in a city. The main idea is that the user will request a taxi from his PDA. This request will reach wirelessly the taxi station, that will forward it to all taxis. Each of them will send a reply with the conditions in which it might serve the request (e.g. the time that it would need to arrive to the user's location). The taxi station will select the best offer and send wirelessly the information to the user, that will receive it in his PDA.



Figure 2. Architecture of the multi-agent system

The architecture of the multi-agent system is shown in figure 2. There are five types of agents:

- *Personal agent* (agente cliente): it is running on the user's PDA. It provides a graphical interface that allows the user to send requests for a taxi and to see the result of his request.
- *Taxi station* (agente centralita): it receives the requests of the users, and forwards them to the individual taxis. It selects the best taxi for the user, and sends the information of this taxi to the personal agent.
- *Taxi agents*: there is one agent of this type for each taxi in town. They receive the user request through

the taxi station. They calculate how long it would take to reach the user's location to pick him up, and send this data to the taxi station.

- *Traffic agent*: this agent has information about the streets that are blocked in the city (e.g. due to traffic jams or works in the road). Taxi agents receive this information at the beginning of their work shift and take it into account when calculating the paths through the city.
- *Visualiser agent*: this agent receives information about the requests (from the taxi station), about the location of taxis and their possible routes (from taxi agents) and about blocked roads (from the traffic agent), and shows this information in a graphical interface.

As you can see on figure 3, there are two containers on the system. The *main-container*, which is running on a standard PC, holds the taxi agents (3 in that figure), the traffic agent, the taxi station, the visualiser and the basic management agents defined by FIPA (DF, AMS and RMA, which manages the GUI of the JADE platform). The PDA holds another container of the same platform, called *container-1*, in which the personal agent is running.

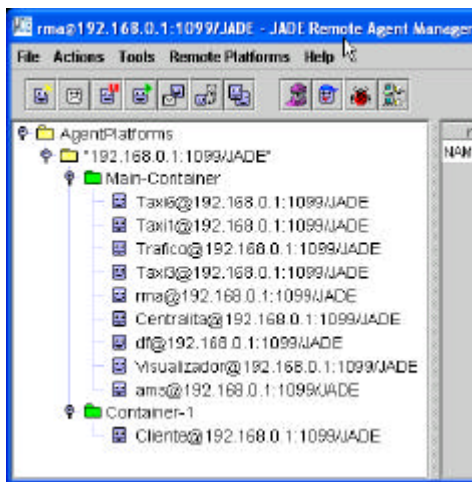


Figure 3. Distribution of the agents in containers

3.2 Requesting a taxi

In this section we will describe how the system works by following all the messages that are exchanged between the agents when a taxi is requested.

First of all, the user is shown a map of the city on the PDA's screen, and he can select with the stylus his present location and the destination of the taxi ride. An example can be seen in figure 4 (the origin is shown

with a yellow square on the top left corner, and the end with a blue square near the bottom right corner).



Figure 4. Initial and final points of the taxi ride.

The personal agent sends the initial and end points of the taxi ride to the taxi station (wirelessly, via Bluetooth). The taxi station sends a CFP (*call for proposals*) to all the available taxi agents (obtained from the Directory Facilitator, DF). Each taxi holds a map of the city, and it applies the A* algorithm to find out the shortest route between his present location and the initial point of the request (taking into account the information on blocked streets provided by the traffic agent). Having made this computation, it sends to the taxi station a proposal with the identifier of the taxi, the time it would take to reach the user, and the route that it would follow to reach this initial point. If the taxi is busy, it does not send any proposal. The taxi station accepts the proposal in which the taxi will arrive sooner to the user's location, and rejects all the other proposals. The selected taxi then calculates the shortest path between the origin and the end points of the ride, and sends this information to the taxi station. Then, the taxi station shows to the user the following information: the taxi identifier, the time it will have to wait for the taxi, the cost of the ride, and the route that the taxi will follow to serve the petition. In figure 5 you can see how the route is shown to the user in the PDA, and in figure 6 you can see the other details of the ride (selected taxi, waiting time and cost).

6 Glossary

API: Application Programming Interface.
FIPA: Foundation for Intelligent Physical Agents.
GPRS: General Packet Radio Service.
JADE: Java Agent Development Environment.
JDK: Java Development Kit.
j2se: Java 2 platform, standard edition.
LEAP: Lightweight Extensible Agent Platform.
MIDP: Mobile Information Device Profile.
PDA: Personal Digital Assistant.
Pjava: PersonalJava.
PPP: Point to Point Protocol.
TCP/IP: Transmission Control Protocol / Internet Protocol
UMTS: Universal Mobile Telecommunications System.
Wi-Fi: Wireless-Fidelity.
WLAN: Wireless Local Area Network.

References

- [1] Wooldridge, M., *An introduction to multiagent systems*, John Wiley Ed., 2002. ISBN 0-471-49691-X.
- [2] AgentCities: <http://www.agentcities.org>.
- [3] Foundation for Intelligent Physical Agents: <http://www.fipa.org>.
- [4] JADE-LEAP: <http://sharon.cslet.it/project/jade>.
- [5] Caire, G. LEAP 3.0 User Guide, TILAB (see [4]).
- [6] Plugfree 2.0. Available at http://support.fujitsu-siemens.de/DriverCD/Start_GB_LIFEBOOK.htm?uri=/drivercd/_DriverSteuerung/GB/LIFEBOOK/Series/LIFEBOOK_S6010_WinXP.htm&menu=/DriverCD/_1st_Start/GB/MenuListe_LIFEBOOK_Series.htm

Contact

Dr Antonio Moreno
Multi-Agent Systems Group
University Rovira i Virgili (URV), Tarragona, Spain.
Phone: +34-977-559681. Fax: +34-977-559710.
E-mail: amoreno@etse.urv.es