Chapter 23
The Development of Jade Agent for Android Mobile Phones

Yu Weihong and Yan Chen

Abstract At present, with the development of mobile applications from the simple calls to data transmission, the contents of mobile applications have become colorful and the requirements to the mobile software will be higher and higher. Especially, as a consequence of the progressive integration between the wireless and wire-line environments, the need of deploying distributed applications such as multi-Agent systems on handheld devices is becoming more and more important. This implementation will make mobile applications more intelligent and proactive but it is very complicated. In this article, Jade-Leap add-on was used to solve the integration of Jade and Android platform. As an example, one Jade Agent for retrieving train running information was successfully developed on Android platform. During the course of research, some problems, such as the communication among Jade Agents, the interactions between Jade Agent and Android Activity, the interactions among Android Activities and so on were addressed and then solved. The application of Agent into mobile platform will make mobile services more intelligent, proactive and interactive, which will meet the personalized needs of mobile phone users.

Keywords Jade agent · Jade-leap add-ons · Android platform · System integration

Y. Weihong (✉) · Y. Chen
Transportation and Management College,
DaLian Maritime University, Dalian 116026, China
e-mail: yuwhlx@163.com

Y. Chen
e-mail: dlmu_chenyan@163.com

23.1 Introduction

One of the most widely used mobile OS these days is Android. Android is a Linux-based operating system for mobile devices such as smart phones and tablet computers. It is developed by the Open Handset Alliance, led by Google, and other companies. Its software bunch comprises not only operating system but also middleware and key applications. Android is open source, meaning that manufacturers don't have to pay Google to use it, and that they're free to modify it. This means that it's used in a wide range of hardware varying in price from small budget phones to large-screen high-end handsets.

In 1986, the notion of Agent has been proposed in the book "Society of Mind" written by Minsky who has made many contributions to AI, cognitive psychology, mathematics, computational linguistics, robotics, and optics. Through the development of more than 20 years, the Agent technology has been evolved from theory to implementation. Especially some standard organization, such as FIPA, has played a crucial role in the development of agents' standards and has promoted a number of initiatives and events that contributed to the development of agent technology.

Java Agent Development Framework, or JADE, is a software framework for multi-agent systems. The JADE platform allows the coordination of multiple FIPA-compliant agents and the use of the standard FIPA-ACL communication language in both SL and XML. A JADE platform is composed of agent containers that can be distributed over the network. Agents live in containers which are the Java processes that provide the JADE run-time and all the services needed for hosting and executing agents. There is a special container, called the main container, which represents the bootstrap point of a platform: it is the first container to be launched and all other containers must join to a main container by registering with it.

In the current researches, Jade Agents mainly run on the computers with fixed IP addresses. But with the development of mobile applications from the simple calls to data transmission, the contents of mobile applications have become colorful and the requirements to the mobile software will be higher and higher. Especially, as a consequence of the progressive integration between the wireless and wire-line environments, the need of deploying distributed applications such as multi-Agent systems on handheld devices is becoming more and more important. But it is difficult to implement.

In this article, Jade-Leap add-on will be used to solve the integration of Jade and Android platform. As an example, we develop one Jade Agent for Android platform. This article will elaborate the principles, the methods and the demo, etc.

23.2 The Principle and Methods of Developing Jade Agent for Android Platform

23.2.1 *Bottleneck Issues*

It is difficult to integrate Jade Agent into Android platform. The developers should have a good command of Agent programming mechanisms as well as Android Activity mechanism, Intent mechanism, Broadcast mechanism and so on. Jade and Android are totally different development platforms, especially due to the limitations of hardware normal Jade Agent can not run on mobile devices such as mobile phone or PDA. Therefore, there are some bottleneck issues in the system development:

1. The complete JADE runtime environment has a memory footprint of some Mbytes that cannot fit the limitations of handheld devices.
2. JADE requires Java 5 (or later) while the majority of handheld devices only support CDC, PersonalJava, or more typically MIDP.
3. Wireless links have different characteristics with respect to fixed network such as high latency, low bandwidth, intermittent connectivity and dynamic IP address assignment that must be taken into account properly.

But fortunately for us, Jade-leap add-on was created to solve these problems and allows deploying JADE agents on handheld devices. For the implementation of deploying Jade Agent to Android phones or other mobile services, we must have a thorough understanding of its principle. In addition to that, during the system development, some interactions must be considered, such as the communications among Jade Agents, the interactions between Jade Agent and Android Activity and the inter-operations among Android Activities.

23.2.2 *Jade-Leap Add-on and its Execution Mode*

The Jade-Leap add-on is based on Jade but replaces some parts of the JADE kernel and then a modified runtime environment will be formed. This new runtime environment can be shaped in different ways corresponding to the two configurations (CDC and CLDC) of the Java Micro Edition and the Android Dalvik Java Virtual Machine. Therefore, we can use Jade-Leap add-on to deploy Jade Agent to mobile phones or other lightweight devices.

The JADE runtime environment can be executed in two different modes: “Stand-alone” execution mode and “Split” execution mode.

In the Stand-alone execution mode a complete container is executed on the target device. On the other hand, in the Split execution mode the container is separated into a front-end (meant for running on the embedded device) and a back-end (for running on the host with J2SE) linked together through a permanent connection. The back-end location acts as a dispatcher for message coming and going to other platforms. This execution mode is particularly suited for resource constrained and wireless devices, because it has the following advantages:

1. Split execution is extremely more lightweight.
2. Split execution minimizes the communication over the wireless link.
3. Split execution is faster in the start-up procedure.
4. Split execution can deal with temporary disconnections.

23.2.3 The Steps and Methods of Integrating Jade Agent into Android

For Android project, Jade-Leap add-on provides JadeAndroid.jar for us. This package file includes a lot of jade classes with ad hoc customization for the Android as well as some services oriented Android, such as MicroRuntimeService. During the development, the first step must be including the JadeAndroid.jar library in the classpath of our Android Application project. And then, we can follow the following steps to develop a Jade Agent for Android phones.

Step 1. Declare MicroRuntimeService in Android Application Project

Consistent with the Android architecture, the Jade runtime is wrapped by an Android service. More specifically, the jadeAndroid.jar library includes two service classes: jade.android.RuntimeService and jade.android.MicroRuntimeService that wrap a full container and a split container, respectively. In this article we use a split container that, in general, is the suggested approach when working with mobile devices. So the MicroRuntimeService should be declared in the Androidmanifest.xml thus it can be identified when the system is running. The function of MicroRuntimeService is to configure the Jade Environment and to start or stop the Jade Runtime when required.

Step 2. Bind MicroRuntimerService with Android Activity

Unlike activities in Android, services run silently in background. Because they have higher priority than inactive activities, it is perfect bind them to the application's component and make the application continue run and response the users' action in the background.

So, now we should bind the MicroRuntimerService which we will use with Android Activity at the application start-ups. The binding function is as below:

context.bindService (new Intent (context, MicroRuntimeService.class), serviceConnection, Context.BIND_AUTO_CREATE);

Step 3. Create a Split Container and Start it

Before this, a main container must be started up and running on a platform server which can be reachable from the android device. The reference codes are as below.

jadeServiceBinder.startAgentContainer("MAIN_HOST", MAIN_PORT, new RuntimeCallback < Void > () {   });

Step 4. Start Jade Agent in the Split Container

Now all set, and we can start our Jade Agent in the Split Container in the following way:

jadeServiceBinder.startAgent("NICK_NAME", "CLASS_NAME", null, new RuntimeCallback < Void > () { });

Step 5. Implement the Interaction between Jade Agent and Android Activity

1. Transfer information from Android Activity to Jade Agent

To do this, when we start the Agent from the Activity we can pass the application Context and other objects as parameters to Agent. Thus in the Agent setup() method it can retrieve these parameters.

2. Transfer information from Jade Agent to Android Activity

In the Agent code, when you need to notify something to the Activity you can send a broadcast message with the action defined in the Activity, of course some parameters could be included by using the putExtra() method of Intent. In the Activity we should create a custom BroadcastReceiver instance and use the IntentFilter mechanism to receive the broadcast from the Agent.

23.3 An Example of Jade Agent for Android

In this article, a distributed multi-agent system for train running information retrieving has been developed. The main idea of this system is that one Jade Agent will be deployed on the user's Android phone; the user will request a train (e.g. the arriving time or departure time at a station) from his Android phone. This request will reach wireless the responder Agent and after finishing the processing of the request, the responder Agent will send a reply with the retrieving results to the requester and the results can be shown on the interface of the user's Android phone.

So, this system comprises two kinds of Agent.

1. Requester Agent

It is deployed and running on the user's Android phone. The user can set the query conditions from the Activity layout and then send the query request to the responder Agent.

2. Responder Agent

It is running on a reachable host with fixed IP addresses. It is responsible for receiving the request from the Android phone and search in the train information database according to the query conditions. After finishing that, it can send the results to the requester Agent.

The framework of the system is shown as Fig. 23.1. The system has been developed under Eclipse. Firstly, we developed the Responder Agent which is a normal Jade Agent and has its CyclicBehaviour to receive and reply message.

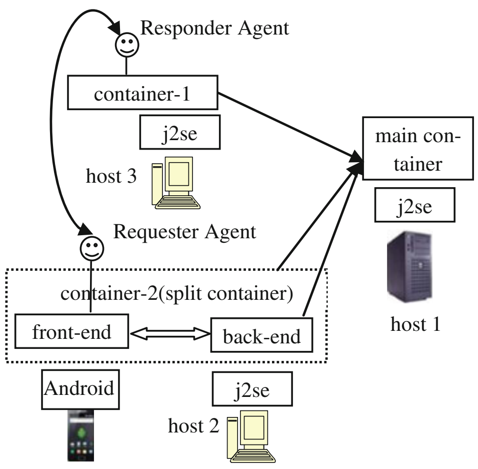


Fig. 23.1 The framework of the system

The more difficult part is to develop the Requester Agent which is a Jade Agent for Android. For that, we created a new Android project and import JadeAndroid.jar.

According to the above principles and methods we made our Requester Agent successfully work on Android. The running steps and results are as follows:

1. Launch the main container with the JADE GUI using the command: java Jade.Boot –gui.
2. Launch a new container and start Responder Agent in it.
3. Run the project as an Android Application, and then we will find a "Train Info" icon show on the Android emulator (as illustrated in Fig. 23.2). This is the Jade Agent we developed for Android, and it is also the frontend in the split execution mode.

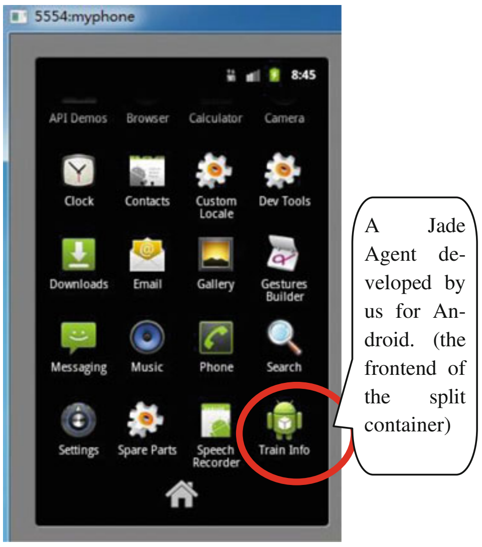


Fig. 23.2 A Jade agent for android was installed on the emulator

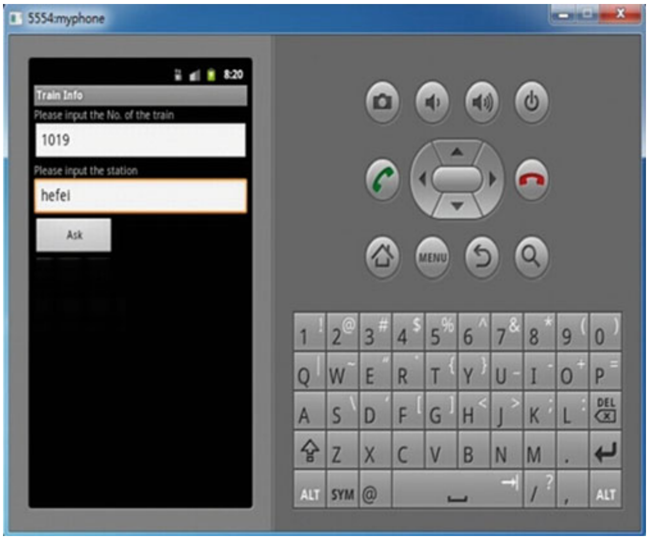


Fig. 23.3 Input the query criterion

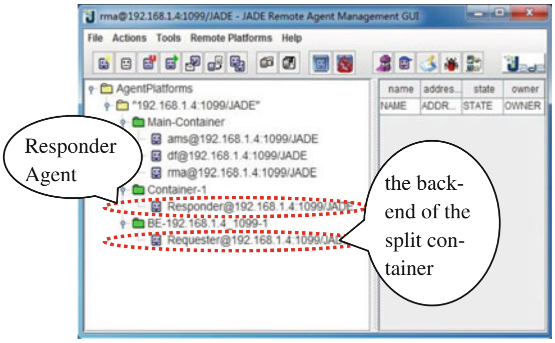
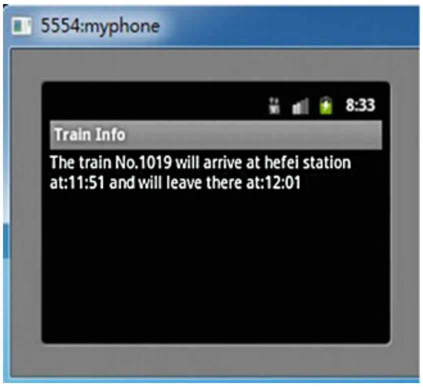

rma@192.168.1.4:1099/JADE - JADE Remote Agent Management GUI
File Actions Tools Remote Platforms Help
AgentPlatforms
"192.168.1.4:1099/JADE"
Main-Container
ams@192.168.1.4:1099/JADE
df@192.168.1.4:1099/JADE
rma@192.168.1.4:1099/JADE
Container-1
Responder@192.168.1.4:1099/JADE
BE-192.168.1.4_1099-1
Requester@192.168.1.4:1099/JADE
name addres state owner
NAME ADDR STATE OWNER
Responder Agent
the back-end of the split container

Fig. 23.4 The change in Jade platform

4. Click the“Train Info”icon,the graphical interface will show as Fig. 23.3.

On this interface we can set the query criterion, for example, input “1019” as the train No. and “hefei” as the name of station. And then, click “Ask” button, we can find that a new container named “BE” has been added and the Requester Agent running on it. Actually, ”BE” is a split container and Requester is its backend. This result is shown as Fig. 23.4.

By using the Jade Agent communication mechanism, the Requester will send the query criterion coming from the Android frontend to the Responder Agent. The responder Agent will send a reply with the retrieving results to the requester and finally the results can be shown on the interface of the user’s Android phone, which is shown as Fig. 23.5.



The train No.1019 will arrive at hefei station at:11:51 and will leave there at:12:01

Fig. 23.5 The result on the phone screen

23.4 Conclusion

In the development of this prototype, JADE-LEAP add-on has been successfully used to implement a personal agent on Android platform. This framework opens the way towards any kind of distributed multi-agent systems, in which personal agents may be smoothly running on mobile devices or other lightweight handle devices and can communicate wirelessly with agents that may provide various services available in different places of the city, such as airports, bus stops, train stations, restaurants and so on. Such application will make mobile services more intelligent, proactive and interactive, which will meet the personalized needs of mobile phone users.