LocaPhone - Location-Aware Group Communication for Mobile Groups

Ulrich Walther and Stefan Fischer

International University in Germany, Campus 3, D-76646 Bruchsal
{ulrich.walther,stefan.fischer}@i-u.de

Abstract. In the context of the Deep Map virtual tourist guide, we present the LocaPhone application which is based on an agent architecture providing virtual group management, group communication facilities over low-bandwidth GSM connections and group member location services using GPS. In the scenario of Deep Map, one typical situation is a group of tourists walking through the city. Since not all of the group members are interested in the same sights, the group might split up from time to time. Still, group members should be able to keep in touch, therefore an application is desirable that on the one hand allows for voice communication, and on the other hand enables the visual localization of other group members. As a result of these thoughts, we developed the LocaPhone application. In this paper, we identify the problems and show how they are solved in our agent-based approach. We discuss both parts of the application (the voice communication and localization) and the agent architecture.

1 Introduction

The driving force behind this paper is an application problem: we would like to provide communication support for mobile virtual groups. Our understanding of such a group is a number of people somehow associated to each other which are moving around a certain area and need or would like to exchange information. Consider a typical example from our DeepMap project, where we, together with many other universities and research institutes under the supervision of the European Media Laboratory, develop an electronic mobile tourist guide for the city of Heidelberg: a group of tourists walking through the city. Since not all of the group members are interested in the same sights, the group might split up from time to time. Still, group members should be able to keep in touch. Just assume that some of the group members are kids, so it is quite natural that their parents would always like to know where they currently are. Keeping in touch may mean two things: on the one hand, there should always be a way to communicate by voice, as it is already possible today using cell phones. On the other hand, it may mean to always know where geographically some other group member is located.

The application that we are going to present in this paper provides the tourist with both features. ”LocaPhone” (for localization and phone) makes use of IP-based voice communication and of GPS-based localization of group members. It integrates both fetaures in one user interface, creating a very powerful communication application for the virtual tourist guide. In this paper we will outline the problems of communication in a mobile environment and then present our solutions in the context of the LocaPhone application. The paper is structured as follows: in the next section, we introduce the reader to the DepMap project as well as to the communication infrastructure available in the city of Heidelberg. Based on this knowledge, we introduce in the following sections the application LocaPhone. In Section 3, we briefly describe our solution to the IP-based voice communication feature. Section 4 introduces the underlying agent architecture for the localization component of LocaPhone and then shows how using this architecture an efficient localization scheme has been implemented. Finally, Section 5 gives an outlook on future work.

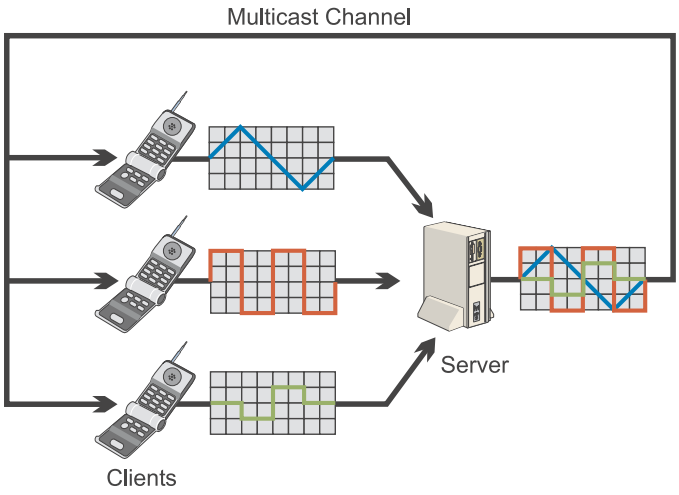

Fig. 1. System architecture. Three conferencing clients, one conferencing server.

2 The Group Communication Component

The group voice communication component was designed to only use 50% of the total bandwidth of a GSM connection (as much as $B_{min} = 4.8$ kbit/s) while still allowing other applications to contact network services such as hotel reservation systems, routing plan servers or natural speech processing servers [11]. If we look at the general case where n participants of a group want to talk to each other, each client must receive the audio signals from the $n - 1$ other members in the group. Even for a small group with $n = 3$ this means that the audio signal must be transmitted at a data rate of $B_{min}/2 = 2.4$ kbit/s (including IP procotol overhead), which results in very bad audio quality considering today's best compression codecs available. For larger groups, audio quality is inacceptable [7, 9]. To achieve an integrated transmission system, IP is used as the network protocol, so both touristic information and voice are carried over IP.

Our solution to the problem of limited bandwidth is to use a dedicated *conferencing server* for each group. Its architecture is shown in Figure 1. The conferencing server receives the (IP/UDP-packetized) audio signals from all participants and computes one single audio signal by mixing together all incoming audio streams. The resulting audio stream is then sent back to the group members by using one IP multicast group, or by using unicast IP. Multicast may e.g. not make sense in MobileIP environments, where multicast routing causes problems. IP packets may experience high overheads and latency when the home agent is far from the best route between mobile host and corresponding host (triangle routing problem). Our server therefore supports both methods.

The resulting audio stream only requires the bandwidth of one single audio stream, thus solving the mentioned problems and offering better audio quality. As a tradeoff, the delay is increased considerably by longer communication paths and time needed for decoding, mixing, and coding MPEG audio packets. In [11], we show how this additional delay can be minimized in order to still allow for reasonable conference quality.

Since each client receives a mixture of all audio signals (containing the client's own contribution), it is desirable to suppress the annoying echo. Therefore each client manages a frame buffer of past contributions to be able to filter its own echo from the incoming audio signal. Voice compression at very low bitrates results in fundamental changes of mathematical properties of the signal (like frequency, phase shift), therefore filtering is difficult in that case. We do not focus on this issue in this paper, but refer the interested reader to [11].

The conferencing client uses the audio hardware to capture the audio signal and to playback the received audio signal concurrently. The captured audio signal is divided into packets of the same size, then each packet is compressed through an MPEG-4 audio encoder [8] and the header of our protocol is prepended. These packets are sent to the conferencing server. On the other side, the MPEG-4 compressed audio frames that are received from the server are decoded, the client's echo is cancelled by filtering the signal from the client's frame buffer, and the resulting signal is played back through the audio hardware. The client is implemented in Java using the Java Media Framework [3] for access to the audio hardware, and using a Java Native Interface [4, 2] for accessing the MPEG-4 codec.

3 The Navigation and Location Detection Component

To be able to compute the user's location, a Global Positioning System (GPS) device is integrated into the virtual tourist guide hardware. Since May 1st 2000, the Selective Availability (SA) that randomly mutilated GPS signals to decrease precision for civil users was switched off, enabling non-differential GPS devices to operate at an accuracy of about 10 meters in average [1].

3.1 Location Agent

Currently we only use GPS devices as the underlying hardware that provides location information within the project environment, but other kinds of sensor devices (like magnetic compasses, additional in-house location devices) may be added due to the abstract handling of location information provided by the VRMI ("Where Am I") agent. VRMI incorporates a Java GPS driver that automatically detects and programs connected GPS hardware, and the VRMI agent functionality that provides several types of location reporting and polling.

Since May 1st 2000, the Selective Availability (SA) that randomly mutilated GPS signals to decrease precision for civil users was switched off, enabling non- differential GPS devices like we use it to operate at an accuracy of about 10 meters in average. Thus, the size of the mobile device is reduced, since the differential reception and processing parts are not needed anymore [12].

GPS receivers send the current position, number of satellites in view, accuracy and precision that can be expected from the measurement and many other information as messages in the standardized NMEA0183 format [10]. The receiver is connected to the wearable computer via a serial communication line (`COMx` in the Windows world or `/dev/ttySx` in Unix). For example, the NMEA message

```
$GPVTG,175.4,T,175.2,M,004.3,N,0007.9,K
```

means that the current true course of the user is in direction 175.4 degrees, the magnetic course is 175.2 degrees, and the user's speed is 4.3 knots or 7.9 km per hour. Due to the asynchronous way GPS devices send information, it is unpredictable when a certain information will be updated next, therefore to each data structure that represents a certain type of GPS data, we added a counter that represents the age of that information. Thus applications can decide up to which age a certain kind of information is still usable, or if it should be excepted from further processing instead of using imprecise values.

To enable access to serial communication lines, the GPS driver uses the `javax.comm` library that is available as an extension to the Java Development Kit [5]. The design of the NMEA message parser uses the ability to reflect on Java classes (`java.lang.reflect`). With Java reflection, it is (among other things) possible to enumerate the members and types of classes. Using that feature, we are able to easily specify the syntax of NMEA messages. This is an advantage, since GPS manufacturers tend to add proprietary NMEA messages to enhance functionality; so when using different GPS receivers it may be useful to include their definitions of additional proprietary messages.

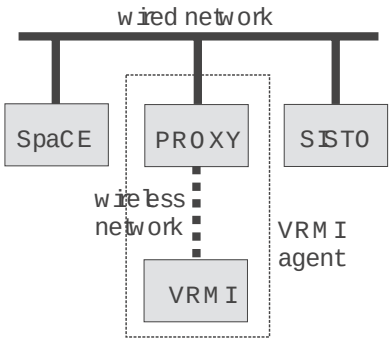

Fig. 2. The VRMI and Proxy agents within the Deep Map agent architecture

For location-aware applications, it is quite interesting and important to know if a user resides in a set of given regions, or if a user stands still for more than a certain period of time, which may indicate his interest in objects located at that position. Therefore, the VRMI agent supports location polling and also subscription to *distance-based* and *time-based/periodic* location reporting [6]. In the first case, the agent reports the new user location when a given distance threshold is reached. In the second case, it reports the user location in a periodic manner, or at a specific point of time. These mechanisms were used to realize the following two types of triggers:

- *region-based trigger*, where the agents sends a trigger to the client agent if the user enters or leaves a polygonal region, and
- *time-based trigger*, where the agents sends a trigger if the user resides within a given distance threshold for longer than a specifiable amount of time.

Since the mobile users will mostly be in areas that are only covered by GSM networks, we must take into account the 9.6 kbit/s bandwidth limit, of which we only want to use a small fraction for location reporting to fulfill the location multicast to the group members. Therefore we decided to split up the location agent into two functional units, namely the PROXY and VRMI units (see Figure 2). The SPACE and SISTO agents shown in addition are further agents within the Deep Map agent architecture and are not related with the LocaPhone application, but provide inference and user-context dependant services to other applications.

The PROXY unit resides on a server in the wired network implementing the agent interface and processing all requests for location information for specific users. Therefore no bandwidth from the wireless connection is wasted with client requests and inter-agent communication. VRMI incorporates the GPS driver and interfaces with PROXY to synchronize their state. In order to minimize the number of necessary location update messages between VRMI and PROXY on the wireless network, we use an optimised protocol that is based on location prediction. In short, both PROXY and VRMI try to predict the future movement and VRMI only sends a location update to PROXY if the prediction does not lie within certain accuracy bounds defined by the user requests being processed by PROXY. The better the prediction simulates the real movement, the less location update messages have to be sent and the more bandwidth on the wireless network can be saved.

3.2 The LocaPhone application

The combination of those two functional entities described in the previous sections results in an easy-to-use phone with built-in user location, the *LocaPhone*.

By using the VRMI agent functionality, the LocaPhone application shows the members belonging to the same tourist group on a city map, thus allowing to e.g. safely trace one's

Fig. 3. Graphical User Interface of the LocaPhone application

children on their way to the city while being able to talk to them using the group communication functionality described in the previous section. All agents that belong to the same tourist group subscribe to one agent group using the Group Management agent (GA) (see Figure 4), therefore LocaPhone can request the logical agent addresses of the other members' VRMI agents from GA. So LocaPhone is able to retrieve location information about all other members in the same group. The required map that should be displayed is requested at one or more map agents (MA); in the shown example, we have to map agents that run on different servers, but it would also be possible to have a small-scale map agent that runs on the mobile device itself.

Figure 3 shows the graphical user interface of the LocaPhone application showing three tourists in Heidelberg. Each tourist is displayed in a different color, the ellipses show the estimated user location area as computed by the GPS driver. Smaller ellipses on the display denote more accurate positions. By clicking on a user's ellipse on the map display, his current position and direction are indicated. On the upper right, the list of members belonging to the user's group is displayed, and the user can start talking to the group by enabling the `Enable Phone` checkbox.

The map is calculated by a map agent that takes the Gauss-Krueger coordinates of the map center as input and returns a bitmap of the area. Since the map agent is still under development, we used a dummy agent that retrieves maps over the internet using the `www.shellgeostar.com` site.

4 Conclusions and Outlook

In this paper, we presented a group communication and a location component thatcan be used to realize the LocaPhone application. LocaPhone provides group member location and voice communication over low-bandwidth GSM connections, which are both useful for groups of tourists. We introduced an agent architecture for the realization of the location and group management services, and a centralized audio server for the implementation of the group communication component. The Java implementation shows the feasibility of our approach.

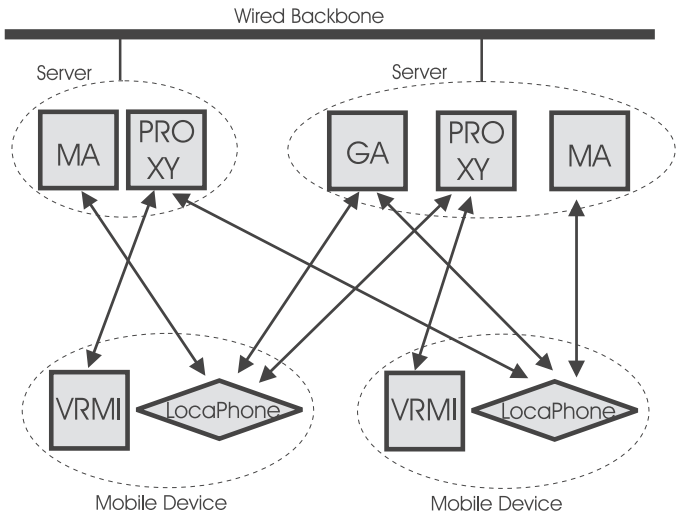

Fig. 4. Two mobile devices running LocaPhone and VRMI, and two servers with VRMI agent proxies (PROXY), Map Agents (MA) and the Group Management Agent (GA)

In the future, we plan to investigate the situation where we have several hot spots in the city that are covered by wireless LAN, whereas the other areas still provide connectivity through the GSM network. To extend the limited range of wireless LAN connections, we currently design an ad-hoc routing layer that may be suitable to provide network connection in situations where the density of tourists between wireless LAN access points is high enough.

5 Acknoledgements

This project is funded by the Klaus Tschira Stiftung gGmbH, Heidelberg.

References

1. U. Coast Guard Navigation Center. Global Positioning System Standard Positioning Service Specification. 2nd Edition, June 1995.
2. R. Gordon and A. McClellan. *Essential JNI: Java Native Interface*. Prentice-Hall, Englewood Cliffs, NJ 07632, USA, 1998.
3. R. Gordon and S. Talley. *Essential JMF: Java Media Framework*. P T R Prentice-Hall, Englewood Cliffs, NJ 07632, USA, 1999.
4. JavaSoft. Java Native Interface Specification, Nov. 1996. Release 1.1.
5. Java Development Kit 1.2. Sun Microsystems, 1999.
6. A. Leonhardi and K. Rothermel. A comparison of protocols for updating location information. Technical Report TR-2000-05, Universität Stuttgart, Fakultät Informatik, Germany, Mar. 2000.
7. MPEG4. Very low bitrate audio-visual coding - Part 3: Audio. *ISO/IEC 14496-3*.
8. MPEG4. Very low bitrate audio-visual coding - Part 5: Reference Software. *ISO/IEC 14496-5*.
9. MPEG4. MPEG-4 Audio verification test results: Audio on Internet. *Atlantic City*, October 1998.
10. NMEA 0183. Standard for Interfacing Marine Electronics Devices.
11. U. Walther and S. Fischer. Group Voice Communication for Mobile Environments. In *Proc. IEEE International Conference on Computer Communications and Networks (ICCCN)*, 2000.
12. D. Wilson. A Comparison of differential and non-differential GPS horizontal accuracy. GPS Accuracy Web Page at http://users.erols.com/dlwilson/gpscomp.htm, 2000.