Experiences Using a Dual Wireless Technology Infrastructure to Support Ad-hoc Multiplayer Games

Hartmut Ritter
Freie Universität Berlin, Germany
Takustraße 9
14195 Berlin
+49 838 75214
hritter@inf.fu-berlin.de

Thiemo Voigt
Freie Universität Berlin, Germany
Takustraße 9
14195 Berlin
+49 838 75210
voigt@inf.fu-berlin.de

Min Tian
Freie Universität Berlin, Germany
Takustraße 9
14195 Berlin
+49 838 75209
tian@inf.fu-berlin.de

Jochen Schiller
Freie Universität Berlin, Germany
Takustraße 9
14195 Berlin
+49 838 75210
schiller@inf.fu-berlin.de

ABSTRACT

Multiplayer games for mobile wireless devices are becoming increasingly popular. Many modern devices such as PDAs or mobile telephones are equipped with Bluetooth interfaces. Bluetooth facilitates the formation of ad-hoc networks thus enabling users to set up multiplayer games spontaneously. On the other hand, Bluetooth has strong restrictions in terms of transmission range and the number of users per piconet. We deploy additional infrastructure in order to overcome these shortcomings. This infrastructure uses a second wireless technology that sets up and maintains a consistent view of a distributed game by exchanging information about the status and location of players. We set up an experimental testbed consisting of single-board communication devices developed in our lab and Bluetooth-equipped PDAs.

Our experiments show that the proposed infrastructure is able to extend the range of games beyond the coverage of a Bluetooth piconet. Furthermore, we are able to track players. Smooth handovers between piconets are also possible. However, due to the current limitations of Bluetooth, handovers are not fast enough to enable highly interactive gaming across different piconets. We present a game that uses this and discuss future application scenarios.

Keywords

Mobile Gaming, Bluetooth, Ad-Hoc Networks

1. INTRODUCTION

Games for mobile wireless devices are becoming increasingly popular. It is expected that by 2006 wireless gaming will generate more than 15 billion dollars in annual revenue worldwide [7].

Today, popular multiplayer games use fixed, preinstalled infrastructure. People play, for example, multiplayer games over the Internet or at LAN parties where the organizer deploys the necessary infrastructure. In this paper, we describe a flexible mobile infrastructure based on two wireless technologies that can be set up and run in an ad-hoc fashion at any place where people meet. A vision not far from realization is people coming together in a subway station and joining an ad-hoc multiplayer game while waiting for the next train to arrive. As a survey by Nokia indicates, nearly 50 percent of the people asked like to play games when waiting [4].

The heart of the presented infrastructure is a single-board communication device developed in our lab that features several communication technologies, including Bluetooth, Ethernet, a CAN bus and communication over 433 MHz.

Bluetooth is attractive for ad-hoc gaming since it is both available on modern PDAs and mobile telephones. Furthermore, Bluetooth facilitates the formation of ad-hoc networks thus enabling users to set up multiplayer games spontaneously. The 433 MHz RF modules on the other hand allow long-distance communication and support communication between equal peers.

We present experiments demonstrating that due to the current limitations of Bluetooth, handovers are not fast enough to enable highly interactive gaming across different piconets. As an example of the types of games our architecture can support, we discuss the transfer of a typical Swedish game to the presented infrastructure.

The outline of this paper is as follows: In the next section we give some technical background on the used hardware and communication technologies. In Section 3, we present the system architecture of our infrastructure supporting ad-hoc multiplayer games. Results of an evaluation in our testbed are shown in

Section 4 where we also present a game that can be nicely mapped onto our infrastructure. After discussing related work in Section 5, we conclude with some future work.

2. BACKGROUND

As part of an infrastructure for research in the area of mobile, context-aware computing and ad-hoc mobile gaming, we built up our own prototypes of embedded devices equipped with different network technologies. We call these devices Embedded Web Server (EWS module) as we implemented in another project a simple HTTP server on the controller, though in this paper we do not use the web server functionality [12].

In the following sections, we give a description of the hardware and software features of these EWS modules.

2.1 The Embedded Web Server Module

The Embedded Web Server module basically consists of a core controller (Motorola 68HC912) and I/O peripherals as well as different wireless and wired network connectors. Figure 1 gives an impression of the overall view.

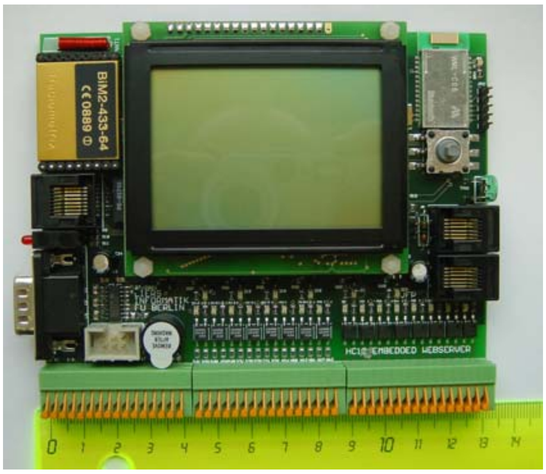

Figure 1: Picture of the Embedded Web Server module

On the upper left hand side resides the 433 MHz-module used for long-range interconnections between the EWS modules. On the opposite side the Bluetooth module can be seen. Hidden behind the display resides the controller. It communicates over serial connections with the 433 MHz and the Bluetooth module. All other on-board peripherals (I/O-ports on the bottom, serial connector and Ethernet on the left side, CAN bus connectors on the right) are not needed for the ad-hoc multiplayer game infrastructure described in this paper. Thus, the hardware needed to set up the infrastructure could be reduced to a board not larger than approximately 5x7 cm, when the display is not needed. A small form factor is a necessary precondition for the envisioned realisation, as large installations in a railway or subway station will not be acceptable.

The core Motorola controller coordinates the message passing over the 433 MHz and Bluetooth interfaces and maintains local state information. In future versions it might be possible to export game information like active players, high scores etc. using the Web Server capability that is already implemented. As the technical aspects of Bluetooth and the 433 MHz module determine the overall system architecture, they will be described in the next two sections.

2.2 Bluetooth

In a Bluetooth network the basic network configuration, called a piconet, consists of a maximum of eight devices. In addition, the transmission range of Bluetooth is only about 10 meters. Building bigger networks could be achieved by combining piconets to scatternets, but this requires scatternet support of the involved Bluetooth hardware and firmware. Specifications of scatternet cooperation are not finished at the moment. In addition, covering a large area using scatternets would require a minimum device density throughout this area, with the risk of network partitioning when nodes disappear.

So we chose the solution to hand over the mobile Bluetooth device between different long-lived piconets. Each Embedded Web Server's Bluetooth module acts as a master for its local piconet and continuously scans the environment for Bluetooth-enabled PDAs and mobile phones. Once a connection is set up, the Bluetooth specification allows measurements of the link quality on the transmission level. The link quality is returned as a value between 0 (bad) and 255 (good). It is described later how these measurements are used.

All the described Bluetooth capabilities (inquiry/scan, piconet formation, link quality measurement, connection management) are provided on board of the Bluetooth chip that can be seen on the upper right corner of Figure 1. In addition to a fully functional Bluetooth stack running as pre-installed firmware, a small user-space program can be downloaded onto the Bluetooth module. This program accesses all Bluetooth functionality and in addition it communicates over a serial line with the core Motorola controller of the Embedded Web Server. The link quality measurements and the resulting connection management are handled by such a user-space program that runs on the Bluetooth module. Thus, a clear separation of tasks could be achieved on the lowest level of the infrastructure. The core Motorola controller only has to coordinate messages and to manage the game logic, whereas all Bluetooth-related, time-critical actions are handled directly on the Bluetooth chip.

2.3 433 MHz

We use 433 MHz RF modules as a second, complementary wireless technology. Whereas Bluetooth is well known in the networking area, 433 MHz RF modules are more common as industry standard for remote control for automotive and indoor applications. As opposed to Bluetooth, 433 MHz RF allows larger transmission ranges of up to 50 m and does not specify MAC or higher layers. Thus, the network formation is variable and the number of devices in a network is in theory unlimited.

The 433 MHz RF module can be seen on the upper left corner of Figure 1. It communicates directly with the core Motorola controller using a serial line. We derived the addresses of the modules on the MAC layer from the Bluetooth device address, as there are no built-in unique MAC addresses with 433 MHz modules. This allows the Embedded Web Server modules to communicate with each other over longer distances using the 433 MHz modules.

3. SYSTEM ARCHITECTURE

Our goal is to develop an architecture that supports ad-hoc multi-user games on mobile devices such as PDAs and mobile telephones. The architecture should require small, cheap infrastructure only and it should be possible to set up the infrastructure very quickly and in an ad-hoc fashion. We envision for example a place like a university, an subway station and other places where people meet and possibly wait for transportation, the beginning of a lecture etc. In these scenarios, spontaneous group formation will take place if it is supported by some technical means. For example, the system could provide high-scores, a gamer's archive allowing persistent nicknames, activity control or cheating prevention.

Based on the possibilities of the hardware described in the previous section we developed a system for gaming support that will be described in this section.

3.1 System Overview

For the envisioned kind of ad-hoc games, Bluetooth is very attractive since building small ad-hoc Bluetooth networks is straightforward. Furthermore, most modern PDAs and mobile telephones provide Bluetooth support. However, the transmission range of Bluetooth is about 10 meters and thus quite short. As argued in section 2.2, Bluetooth scatternet support has some major drawbacks. Therefore, we choose a different approach and set up what can be roughly defined as a roaming infrastructure: In order to overcome the problems of short-range Bluetooth piconets, we distribute several EWS modules as described in section 2.1 across the field. We use the 433 MHz RF technology on the EWS module as a second wireless technology. Bluetooth is used to transport gaming data, whereas the 433 MHz technology is used to maintain a global view of the game, to track the players and to exchange gaming data between the piconets. Another usage of the 433 MHz technology is to find out whether players have left the game. This dual wireless technology infrastructure offers several interesting possibilities:

- The entire infrastructure needed is cheap, flexible and easy to set up in an ad-hoc fashion.
- The infrastructure itself is mobile, there are no wired links between the involved devices.
- Since the transmission range of the 433 MHz RF modules is larger than the range of Bluetooth, the distance between two adjacent EWS modules can be larger than the transmission range of Bluetooth. Therefore, the field can be enlarged without the need to form Bluetooth scatternets.
- The full bandwidth that Bluetooth offers can be used to transport gaming data, only data that has to be routed to another piconet passes the 433 MHz RF link.

The Bluetooth chips on the Embedded Web Server modules act as the masters of the Bluetooth piconets around them. The Bluetooth chip informs the EWS module when a connection to a player is broken which means that the player has left the piconet. When the player is within coverage of the next EWS module, the Bluetooth device of the player is inquired by the Bluetooth chip of this EWS module. The latter informs the adjacent EWS modules about the newly found player using the 433 MHz technology. If an EWS module looses a player and neither finds her again nor does it receive a notification that an adjacent EWS module has found that player, it can assume that the player no longer participates in the game. Obviously she has either left the game or turned off her Bluetooth device.

3.2 System Details

When making some preliminary experiments with Bluetooth, it turned out that our Bluetooth chip requires several seconds until it detects that a slave is out of range. We therefore implemented another approach in which the Bluetooth chip on the EWS module continuously measures the link quality of the connections to its slaves. If the link quality falls beyond a certain threshold, the master breaks the connection to the slave and the core controller is informed. This has the additional advantage that it allows to shape the range of the Bluetooth piconet and to adapt it to the physical topology. We also defined two other link quality thresholds that indicate if the link quality of a connection to a slave is high or low. When the link quality of a connection passes a threshold value, the Bluetooth chip sends a message to the core controller of the EWS module.

As mentioned above, EWS modules inform their neighbours when they find a player. Note that the EWS module could also inform its neighbours when the Bluetooth module breaks the connection to a slave. However, it turned out to be simpler to inform neighbours when a new slave (player) is found only.

For communication between the 433 MHz RF modules we use a very simple ASCII-based protocol. The messages contain a header consisting of the sender and receiver ID, a message type, the packet length and the payload. The protocol implementation supports both reliable and unreliable messages. For our purposes we have defined several message types (e.g. FOUND when a new player is found). Thus, it is sufficient that the payload contain the Bluetooth address of the affected player only.

In our current implementation the EWS modules are numbered and have a priori knowledge about their neighbours which is used for direct neighbour-to-neighbour communication. We leave the definition of a true zero configuration protocol as future work. Note that this is not a real problem. When players place the EWS modules in the field, they just have to make sure that they place the modules according to their consecutive numbering (Figure 2 shows an example).

4. EVALUATION AND DISCUSSION

The infrastructure supporting ad-hoc multiplayer games described so far was fully implemented and evaluated in the context of our university building. This allows to get practical experiences and to uncover difficulties. We see it as a platform for future research of different kinds of games. One example of a game application using this infrastructure is being described in this section.

4.1 Testbed

Our testbed depicted in Figure 2 consists of four EWS modules that are placed inside the offices of our lab members. During the experiments the affected four office doors are closed. The distances between adjacent offices are 12 m (EWS 1 in office 1 to EWS 2), 15.3 m (EWS 2 to EWS 3) and 16 meters (EWS 3 to EWS 4). Fortunately, this is farther than the transmission range of Bluetooth but less than the double of the transmission range. As expected, 20 meters is in the transmission range of the 433 MHz. The baud rate of the 433 MHz transmission is set to 19200 in our experiments. As client device we use an iPAQ 3970 PDA equipped with Bluetooth, running Windows CE 3.0.

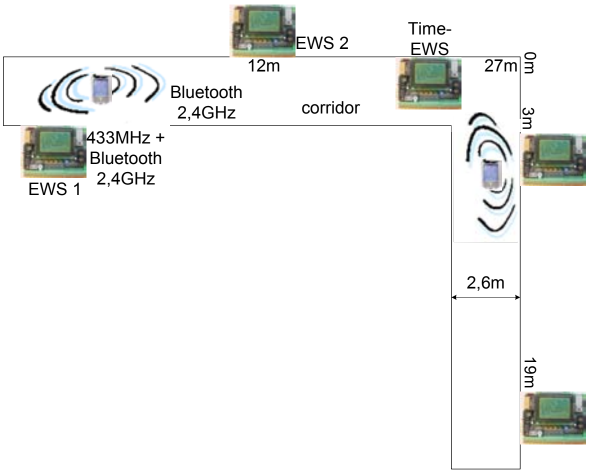

Figure 2: Testbed

In order to collect distributed timing information, we use a fifth EWS module, called time-EWS, this time without a Bluetooth chip. It serves for fine-granular logging of the distributed system's behaviour. The other EWS modules inform the time-EWS by sending messages over 433 MHz RF technology. The message type depicts the nature of the event. Time differences between events can be computed by calculating how often the timer interrupt between two successive events is triggered. This timer interrupt is triggered by the hardware every 65 ms.

4.2 Handover

In this experiment, we measure the Bluetooth handover time, which we define for the purpose of this paper as the time difference between the teardown of the connection by the first EWS module and the moment the mobile client is found by the second EWS module. This handover time is of extreme importance to determine the types of games that can be played on our architecture. Before conducting this experiment, we validated that when a player moves from one EWS module to the next, it is in the transmission range of the second EWS module before the connection is disrupted by the first EWS module because the quality of the connection falls beyond the threshold as described in Section 3.2.

In this experiment, an EWS module notifies the time-EWS when it disrupts the connection to a client. When an EWS module finds a client, it also sends a message to the time-EWS but this time another type of message. The time-EWS can now compute the handover time, i.e. the time difference between the arrivals of the two messages. Note that a Bluetooth slave can belong to one piconet only; thus, it cannot be found by another Bluetooth master before it has left its current piconet.

Our test runs resulted in an average handover time of 1.43 seconds. The results were very similar for the three sections of our setup. We also redid the same experiments using a laptop instead of the iPAQ and achieved similar results.

We expect that the results would be similar using other architectures built on Bluetooth that extend the range beyond a single piconet. For example, Kansal and Desai have simulated handover times in a Bluetooth based public access infrastructure [6]. They determined the inquiry time as 1.35 seconds.

4.3 Application to Gaming

Due to the large handover time, we do not assume that highly interactive multiplayer games are feasible using our architecture. Unless, of course, handovers take place during non time-critical moments that are part of many games, for example before the start of a racing game or when switching between game levels. Note that, since our EWS modules monitor the signal strength, we have the possibility of warning players when they move into a zone with weak connectivity and may thus trigger a handover.

We now present a type of game which we are currently adapting to our infrastructure. In Sweden, a very popular game played at events such as weddings is "tipspromenad" which could be translated to "walkaround quiz". In this game, postings with one multiple choice question on each posting are set up at different places (the game is often played outside and the postings are set up on trees, fences, the backside of houses etc.). People now walk around, looking for the postings and setting a cross for each right answer on an extra sheet of paper which they hand in to the referee when they have answered all the questions.

We believe that our infrastructure is suitable for this game and would make it more interesting. The questions would be shown on the display of the EWS module and via Bluetooth players would transfer the answer to the EWS module which could via 433 MHz RF communication transfer the answer to a central server (via some neighbours if not in transmission range of the server). Using 433 MHz technology, the EWS module could find out if players leave the field before they are done with all questions. This way, for example, players leaving the field in order to sneak into the library could be detected. A feature that makes the electronic version of the game more interesting is the maximum number of devices per piconet imposed by Bluetooth. The EWS module could exchange their current number of active users and this way, users could get advice if they should move towards the left or right EWS module.

Note that the web server facility of the EWS module could be used in at least two ways: First, to make the results and even current rankings available. Second, to allow people to download the game on their PDA. Thus, people could just come with their PDA, download and install the game and start playing immediately.

5. RELATED WORK

Previous work in the area of networked multiplayer games often has focused on the issue of network delay. Network delay impacts the players' gaming experience and makes it hard to achieve a consistent system state. Different approaches have been developed to cope with the latter problem, for example, dead reckoning schemes and local presentation delay [9]. An example of local presentation delay called bucket synchronization has been implemented in MiMaze [3]. Pantel and Wolf have studied the impact of delay on the user experience and found that delays up to 100ms are acceptable for car racing [10]. While in the scenarios above, transmission and queuing delays exist during the whole duration of the game, in our architecture handovers happen every now and then but not continuously during the game. The impact of these on the gaming experience surely depends on the type of game and the phase of the game during which the handover occurs. As discussed previously, our infrastructure makes it possible to warn the players when a handover is about to occur.

Mauve et al. have proposed proxy architectures for networked games which are expected to help with congestion control, robustness, provisioning of fairness and reduction of the impact of delay [8]. In our architecture the EWS modules could implement these tasks if necessary.

Fitzek et al. have also designed an architecture featuring two wireless technologies [2]. In contrast to our work that aims at providing an infrastructure for ad-hoc gaming, games in their architecture are played over commercial infrastructure.

An alternative to building the kind of roaming infrastructure we presented in this paper could be the formation of Bluetooth scatternets. Besides the fact that multi-hop scatternets require a minimum device density in a given area, they are not expected to have a diameter of greater than 5-10 hops [11]. In this case scalability concerns are more critical than with an infrastructure approach.

Kansal and Desai discuss the problem of handover times in larger Bluetooth networks [5]. On top of the Bluetooth HCI interface they propose a new protocol that reduces handover delay. If these mechanisms were commonly available or even part of a new version of the Bluetooth specification, we could use them to improve our infrastructure and to support a broader range of gaming applications. The work of Baatz et al. also deals with the problem of long handover times and analyses interactions with higher-layer protocols [1]. They use special inquiry access codes for faster discovery of Bluetooth access points.

6. CONCLUSIONS AND FUTURE WORK

As its main contribution, this paper has presented an infrastructure supporting ad-hoc multiplayer games. Our infrastructure features two wireless technologies implemented on a single-board device developed in our lab. Thus, we realised a fully mobile infrastructure. It allows cross-piconet coordination and communication and uses the measurements of link quality for keeping track of user position and movements.

While Bluetooth facilitates the formation of ad-hoc networks, its time-consuming inquiry operation leads to long handover times. These handover times restrict the types of games this architecture is suitable for. We are currently implementing a game that does not suffer from these handovers but we also aim at studying the impact of these handover times on users' experience with games with stricter real-time requirements. We are also interested in analysing multiplayer games to see which types of games offer suitable phases during which delays caused by handover times are acceptable. We see a great potential for performance improvements in the coordination between communication constraints in the real world and their representation in events of a game in the virtual world.

7. ACKNOWLEDGEMENT

The authors want to thank Achim Liers who developed the EWS as well as the students who implemented the EWS software. Special thanks to Enrico Köppe for valuable help during the implementation of our infrastructure.

8. REFERENCES

[1] Baatz, S., Frank, M., Göpffarth, R., Kassatkine, D., Martini, P., Schetelig, M. and Vilavaara, A.: *Handoff Support for Mobility with IP over Bluetooth*. 25th Annual Conference on Local Computer Networks (LCN), Tampa, USA, 2000.

[2] Fitzek F., Schulte G. and Reisslein M.: System Architecture for Billing of Multi-Player Games in a Wireless Environment using GSM/UMTS and WLAN Services. Netgames 2002, Braunschweig, Germany, 2002.

[3] Gautier L. and Diot C.: *Design and Evaluation of MiMaze, a Multiplayer Game on the Internet*, International Conference on Multimedia Computing and Systems, Austin, TX, USA, 1998.

[4] Graham, T.: *The Future of Mobile Gaming*. SmartPhone Conference, 2000. Can be found at: http://www.gsmworld.com/technology/applications/entertainment.shtml

[5] Kansal, A. and Desai, U.: Motion *Constraint Based Handoff Protocol for Mobile Internet*. IEEE Wireless Communications and Networking Conference (WCNC), March 2003.

[6] Kansal, A.: *A Handoff Protocol for Mobility in Bluetooth Public Access*. Masters Thesis. Indian Institute of Technology Bombay, 2002.

[7] Leavitt, N.: *Will Wireless Gaming be a Winner?* IEEE Computer Magazine, January 2003.

[8] Mauve, M., Fischer, S. and Widmer, J.: *A Generic Proxy System for Networked Computer Games*. Netgames 2002, Braunschweig, Germany, 2002.

[9] Pantel, L. and Wolf, L.: *On the Suitability of Dead Reckoning Schemes for Games*. Netgames 2002, Braunschweig, Germany, 2002.

[10] Pantel, L. and Wolf, L.: *On the Impact of Delay on Real-Time Multiplayer Games*. International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2002), Miami Beach, Florida, USA, 2002.

[11] Raman, B., Bhagwat, P. and Seshan, S.: *Arguments for Cross-Layer Optimization in Bluetooth Scatternets*. Symposium on Applications and the Internet (SAINT'01), 2001.

[12] Information about the Embedded Web Server can be found at: http://www.scatterweb.net