A Communication Methodology for Interactive Location-Based Mobile Games

Kristian Nybom[1,2], Janne Kempe[1], M. Mohsin Saleemi[1,2], Jerker Björkqvist[1], Johan Lilius[1]
[1]Department of Information Technologies, [2]Turku Center for Computer Science,
Åbo Akademi University, Finland, {firstname.lastname}@abo.fi

Abstract
In this paper, we study how to implement interactive services when using a DVB-H broadcast channel combined with a point-to-point channel, such as 3G or GPRS. We study the problem in the context of a location based game. We explore design issues and problems related to the scheduling of contents in the game and present an architecture for the game server supporting this. We conclude that most of the problems involved with our approach can be expressed as the problem of defining delivery deadlines for a scheduling algorithm.

1. Introduction

With the evolution of handheld devices, positioning techniques and new transmission technologies, a new class of location-based mobile applications (Dix et al., 2000) has been made feasible and has become an evolving research area. These mobile location based games involve wireless networks, navigation techniques, movement and location of the players, and hypermedia content that are used by the players while playing game. Details about these kinds of games can be found in (Klante et al., 2005; Drab and Binder, 2005; Boll et al., 2003). These publications do not discuss communication methodologies or the server architectures. In our interactive location-based mobile game, we use a new wireless broadcast technology with the old theme of traditional location-based games. This approach is innovative and gives the opportunity to explore the concepts and issues involved in this new converged environment. In this paper, we discuss the challenges and problems that are involved in designing this game due to the usage of wireless broadcast technologies. The paper includes scheduling problems, an implementation sketch of the server architecture and the overall interactive communication setup to illustrate how the broadcast channel is used in this interactive mobile game.

The rest of the paper is organized as follows: Section 2 describes the motivation behind the game scenario. Section 3 provides the definition and categories of interactivity in this kind of system. Section 4 presents the overview of the communication system. In Section 5, we present the game platform architecture and the communication methodology. This section represents the overall implementation sketch. Finally, we describe future work and conclude the paper in Section 6.

2. Motivating Game Scenario

Abot is a multiplayer interactive mobile game in which the movement and location of the players trigger all the actions. We use GPS for obtaining the location information. The communication between the terminals and the game server is done using two different channels. 3G or GPRS is used as the uplink to the server, to send information from the terminals to the server using an IP connection and a wireless broadcast medium (DVB-H) to communicate with the players. The real physical environment serves as playing area for the game. The game provides bridging of physical locations and objects to a virtual map on players' handheld screens. The game has two different kinds of tasks for the players. The players can collect bots, which are virtual containers that are dispersed on the game map by the server. Bots contain hints, points, equipment etc, which are useful in solving quests. Quests are the second kind of task in the game. Quests in their simplest form involve finding locations by solving puzzles, and they can also be constrained in time. Quests form the story of the game. The players in the game are equipped with handheld terminals having mobile phone networks and capable of receiving wireless broadcast data. Additionally, the players are equipped with a GPS device to get accurate positions on their mobile phones.

The novelty of the game is that we use DVB-H, a broadcast medium to deliver data to the players. Our motivations for using DVB-H are the following. Firstly, DVB-H is optimized for handheld terminals and offers much higher data rates that cellular networks like 3G or GPRS. In the game design we want to explore this possibility to give a more immersive experience for the players, but this is not the topic of this paper. Secondly, DVB-H is a relatively new technology. We want to use the context of a location based game to explore this technology in a converging environment where both cellular and broadcast technologies are used. In this way we hope to expose new research issues, and challenges.

3. Definition of Interactivity

Jensen (2005) discusses many forms of interactive genres, formats and content in the context of Interactive TV. In this paper we are not so much concerned with different kinds of interactive TV, but more with the technological issues in the implementation of interactivity. Therefore in this paper, an interactive service is defined as a service where the action of one user affects the internal status of the server providing the service, allowing all other users to experience the changes. Using this definition, a web discussion forum, for example, is an interactive service. However, the interactivity in such a service is dependent on the user's actions, i.e. the user has to request some data in order to be interactive. This can be defined as pull-type interactive service. Here our special interest is when the interaction affects a common stream of data to many users. A typical example of such an interactive service is the SMS-based chats in television, where you can send SMS messages to a server, which then broadcasts the messages as a video stream. This can be defined as a push-type interactive service.

We can also define interaction from the point of view of the user. In this case we can distinguish between active and passive interaction. The passive interaction is used for interacting with the server periodically, i.e. sending the location with a fixed interval. The active interaction, on the other hand, is initiated when the player has to make a decision that affects the state of game and other players. An active interaction can for instance be when a player requests some information from the server to perform an action or when a player makes a choice to accept or reject some action that has influence on other players. Example of active interaction in this game is when the player picks up a bot or accepts a quest that is no longer available to other players.

Given these definitions of interactivity and the motivating game scenario we can define three categories of interaction in this type of system:

A. Instantaneous pull-type interaction
B. Predicted push-type interaction
C. Collective push-type interaction

Each interaction poses individual requirements on the implementation. Instantaneous pull-type interaction (A) is sporadic communication which is followed by a near-instantaneous response and typically consumes very little bandwidth. Long response latency in this type of interaction is directly observed as an unpleasant user experience. Predicted push-type of interaction (B) covers the case where the location history or some other user action sequence reveals to the system the most probable near-future communication need of the user. The server should be able to push data to the terminal before the user actually requests it. This creates the same type of user experience that the instantaneous pull-type interaction does, but is used for communication of larger amounts of data (e.g. a video clip which is viewable immediately when requested). The collective push-type interaction (C) is related to multicast data which reflects the current state of the service, which in turn is affected by passive user interaction. This interaction is for instance that the server periodically multicasts data containing the locations of all the users. The SMS-based chat described earlier is also one example of collective push-type service. All these types of interactivity can be implemented into the proposed server architecture.

4. Communication System

For exchanging data between the terminals and the server, an asymmetric communication channel is employed. In the asymmetric communication channel, the terminals use a point-to-point (PtP) link, such as 3G or GPRS, to the server, while the server communicates the data to the terminals using DVB-H multicasting. Using the PtP link, the users are able to inform the game server of their actions and, thus, actively interact with the game. Based on the received messages from the users, the server modifies the state of the game and then broadcasts appropriate data that is needed by the users. Because each terminal is connected to the server through the PtP link, the server has an alternative option to use this link for distributing data. This is covered in more detail in section 5. The communication between the server and the terminals is illustrated in figure 1.

In order to maintain the quality of the user experience in the game, it is critical that every user receives all the data that is intended for that user. Because DVB-H cannot guarantee the delivery of uncorrupted multicast data to all receivers, additional error correction is required in the DVB-H communication channel to ensure the delivery of content. The DVB-H standard specifies the Raptor code for usage in file deliveries (Shokrollahi, 2006) that can be efficiently exploited in this environment. The Raptor code improves the reliability of content delivery in communication systems, at the cost of slightly increased overhead in the transmission.

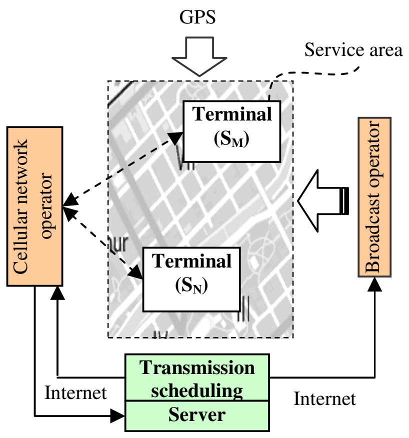

Figure 1. Communication system overview.

Because data is being multicast from the server to the receiving terminals, it is important for the terminals to determine which data are relevant for them and which are not. This is easily implemented with additional header information, embedded in the transmitted game data, allowing the terminals to parse the headers and retrieve only the relevant data.

5. Implementation Sketch

The implementation is realized with a PC connected to the Internet, which is used for delivering IP data unidirectionally to the broadcast operator and bidirectionally to the cellular network operator. The broadcast channel uses UDP and the PtP channel uses TCP. The players connect to the server using the PtP connection and negotiate a subscriber account, which contains all player specific information. In the following sections a player is denoted as a subscriber. The concept of a subscriber is introduced to distinguish the physical player from the limited player specific information maintained on the server. The implementation sketch of the system begins with an overview of the server architecture and continues with a brief overview of the content scheduling methodology.

5.1. Game Platform Architecture

Figure 2 shows the logical block diagram of the server implementation.

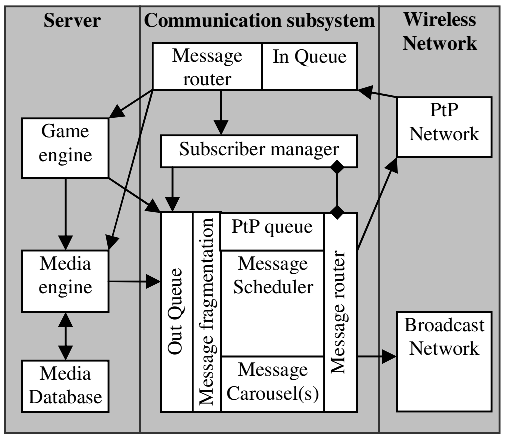

Figure 2. Server logical block diagram.

The game engine is the entity which keeps track of the state of the game. Its main tasks are to update the positions of the subscribers and based on that to decide on what new information each subscriber (or group of subscribers) needs next. This also involves predicting ahead what data a particular subscriber will require in the near future. This is explained in more detail in section 5.3. The media engine is the entity which handles all the objects, in the media database, associated with the service. The media engine produces all messages which contain service objects. In our motivating game scenario such objects can be documents, images, storyline animations done in Flash, video clips, etc. In addition, the game engine may compose dynamically directed live video and audio feeds reflecting for instance the state of the game). The system encourages the subscribers to produce own content to the service by uploading e.g. digital images or video clips.

The subscriber manager keeps track of users or players that are currently using the service interactively. The subscriber assigns correct IP addresses to packets encapsulating data destined to a certain subscriber or group. In addition, the subscriber manager handles congestion control for individual terminals. During login, the memory capacity of each terminal is negotiated to the server and taken into account before transmission, in order to avoid buffer overflow in the receiving terminals.

The message handling in the communication subsystem is realized using message queues for input and output messages, a message carousel, and a message router. Messages may be of any length and have arbitrary due time for delivery. This imposes the need for a pre-emption mechanism for message transmission. In other words, an urgent message must be able to pre-empt a less important message. The solution to this in the presented system is to use a message scheduler.

In addition, we assume that the number of available downstream channels is arbitrary, that they possess different characteristics, and that these should be taken into consideration when the message routes are chosen. In case of a PtP network, the communication is reliable and straightforward. However, a broadcast channel, such as DVB-H, is unidirectional and unreliable. Also, the reception condition for each terminal is different and from this point of view, the subscribers cannot be seen as a homogenous group. Using the broadcast channel for delivering messages to different sets of subscribers, even individuals, requires intelligent decisions on usage of the available bandwidth. Furthermore, since the broadcast channel is unreliable, the message should, if needed, be retransmitted. Retransmission in the server is handled by the message carousel which contains data that is either not yet transmitted or not yet acknowledged by all recipients. Raptor encoded messages in the carousel are not retransmitted but are instead replaced with new encoding symbols. Longer messages need to be fragmented into smaller messages that the scheduler can handle. The fragmentation process preserves the requirements of the original message.

The message router for the output messages receives messages from the PtP queue and the message carousel. The messages are encapsulated into IP packets and routed to the address acquired from the subscriber manager. The predefined network is consequently used for delivering the message.

Input messages are always received using the PtP network. The messages are then routed, based on type of message, to the correct module, which then handles it accordingly.

5.2. Communication Methodology

The elementary idea of the system presented in this paper is that the individual subscriber, equipped with a terminal supporting the system, may affect the content and the state of the service by his or her actions. All messages in the system use the same message header, which contains the unique ID of the recipient, the message length, an identifier for the type of message, and payload data. The message protocol also supports fragmentation.

Messages in the system can be categorized into seven different classes (table 1). Typical for message categories 1-2, related to registration and authentication of the subscriber, are that these messages do not affect other subscribers and that they do not require extensive system resources. Categories 3-4 are used for subscriber interaction. Periodic reporting of the current location corresponds to passive interaction, whereas messages in category 4 are associated with active interaction and requires special attention and follow-up. The messages belonging to categories 5-6 are directly related to the current or upcoming state of the game engine. These messages are divided into individual messages and group messages, because the group message will be transformed into multiple individual messages in case the PtP route is decided to be used. Messages belonging to category 7 are special messages which may consist of e.g. encoded video stream. The concept of a message is still used for category 7 because of the scheduling system.

Table 1. Message classes

Cat	Source	Destination	Message	Route(s)
1	Subscriber	Subscriber manager	Login/logout, registration	PtP
2	Subscriber manager	Subscriber	Login/logout, registration	BC, PtP
3	Subscriber	Game Engine	Location update/ ACK	PtP
4	Subscriber	Game Engine	Content upload / active interaction	PtP
5	Game Engine	Subscriber	Individual service data	BC, PtP
6	Game Engine / Media Engine	Group / All subscribers	Common objects	BC, PtP
7	Media Engine	All	Stream data	BC, PtP

Table 1 indicates that the PtP network is always used as feedback channel and downstream data is delivered using either the broadcast network or the PtP network. Certainly there are several valid alternatives for message routing strategies. The goal for the system in this paper is to define cost functions for all available downstream channels. The messages are characterized according to their length, due time, destination, and interdependence to previous messages (i.e. predefined order). The message characteristics are used as constraints when solving the scheduling problem and minimizing the cost functions. The scheduling is done periodically by the carousel scheduler. The basic algorithm is that the scheduling problem is solved for a set of messages already in the carousel together with new messages in the out queue. The solution to this problem should yield the set of messages to be sent over the PtP network and a batch program for the message carousel, from which data is broadcasted. The solution is assumed to be valid for the rest of the scheduling period. Instantaneous interaction messages can be handled either by an immediate rescheduling or by pre-allocating a fixed bandwidth for this type of messages.

5.3. *Scheduling of Content on Server*

One challenge for this game environment is to provide all user terminals with data that is needed at the particular moment. The assumption is that the user terminal memory capacity is limited, hence the user terminals must be provided with data on demand. Additionally, some of the data sent is dependent of the status of the game; hence it is not available before it is needed. The demand for data is assumed to be predicted by the game server. For the scheduling, the problem is formulated as following.

The game server schedules packets, indexed *p*, to be sent to receiver terminals. Each packet can be either unicast or multicast. Each packet can be sent over a number of distinct communication channels, or networks, denoted with index *n*. The size of the data needed to be sent is given by S_{pn}, where the size is also assumed to be dependent on the network used. In a unicast network (e.g. GPRS), a packet intended for many receivers is sent many times, hence the actual data sent is multiplied in such network. In multicast (e.g. DVB-H) the data can be sent in parallel to many receivers. Each packet is associated with a release-time, earliest time when the packet can be scheduled, T_p^R and a deadline, T_p^D. Each network is associated with a cost per data sent, C_n, and a data rate, R_n. The scheduling problem is formulated as minimizing the total cost for sending the data. The decision variables are when the packet should be sent, t_d, and using which network it should be sent, x_{pn}, where x_{pn} is a binary variable specifying if packet *p* is sent over network *n*. Hence the objective function is

$$\min\left\{\sum_p\sum_n C_n S_{pn} x_{pn}\right\} \tag{1}$$

The constraints for the optimization problem are that the deadlines should be met, $t_p \leq T_n^D$, and the release-time should be met, $t_p \geq T_p^R$. Additionally, we make the assumption that a unicast network (i.e. GPRS) can simultaneously distribute packets intended for many receivers, where as in the multicast network, the packets must be scheduled not to overlap each other. The constraints for scheduling of the multicast networks can be done using disjunctive programming.

The overall optimization problem can be solved using mathematical programming approaches, however, in order to keep the optimization fast and efficient, the problem is broken in to time windows, i.e. the set of indexes *p* for optimization is selected from packets *p* with deadlines T_p^D inside a time window $T_p^D < T_i^W$, where time window, T_i^W, *i* is selected using heuristics. The heuristics is used for ensuring that the maximum size of the optimization problem is small enough (set of indexes *p* is small enough), or the end of the time window T_i^W is not too far in the distance.

By using acknowledgement of received data items, lost data items can be identified and can be resent using the PtP network. This procedure enhances the QoS in the heterogeneous network environment.

6. Conclusions

In this paper we have presented how interactivity can be implemented in an environment where content is delivered using a broadcast medium. We have identified 3 types of interaction that need to be supported by the server system. We have proposed an architecture for the server and also shown how to express interactivity as a scheduling problem. As future work we plan to study how to derive the packet deadlines T_p^D from the structure of the game.

References

Boll S., Krösche J. & Wegener C (2003). *Paper Chase Revisited — a Real World Game Meets Hypermedia.* In 14th ACM conference on Hypertext and Hypermedia, Nottingham, UK.

Dix A., Rodden T., Davies N., Trevor J., Friday A., & Palfreyman K. (2000). Exploiting Space and Location as a Design Framework for Interactive Mobile Systems. *ACM Transactions on Computer-Human Interaction (TOCHI). 7*(3), 285-321.

Drab S.A. & Binder G. (2005). *Spacerace - A Location Based Game for Mobile Phones using Assisted GPS.* In PerGames.

Jensen, J. F. (2005). *Interactive Television: New Genres, New Format, New Content.* In 2nd Australasian Conference on Interactive Entertainment.

Klante P., Krösche J. & Boll S (2005). *Evaluating a Mobile Location-Based Multimodal Game for First Year Students.* In Multimedia on Mobile Devices.

Shokrollahi, A. (2006), Raptor Codes, *IEEE Transactions on Information Theory*, *52*(6), 2551-2567.