Project Avatar — Developing a Distributed Mobile Phone Game

Mattias Andreasson, Andrée Bylund, Syrus Dargahi, Daniel Johansson, Martin Larsson, Bennie Lundmark, Jonas Mellberg, Fredrik Stenh Olle Gällmo, Anders Hessel, Leonid Mokrushin, Paul Pettersson

Department of Information Technology, Uppsala University
P.O. Box 337, S-751 05 Uppsala, Sweden.
Email: {maan4367,anby8843,syda8965,dajo5983,mala1167, belu6251,jome8925,frst0007}@student.uu.se, {crwth,hessel,leom,paupet}@it.uu.se

Abstract. Team Avatar, as the members of Project Avatar have come to be known by, is a group of 4th year computer science students at Uppsala University that have been developing a distributed mobile phone game during the fall of 2005. In this paper we describe the general design and environment of the result of Project Avatar — the game Three Crowns.

1 Introduction

Team Avatar, consisting of eight computer science students at Uppsala University, was founded in the fall of 2005 as one half of the 4th year project course workforce, the other half being Team A-GPS. The task to be completed was to develop a game for mobile phones running over a wireless Bluetooth network. Due to the unspecified nature of the original task and some technical issues regarding the Bluetooth architecture, the task shifted to become the development of a distributed game for mobile phones communicating through a third part, a server accessible from the Internet, as well as communicating over Bluetooth.

The aim of the project course is to give the participating students knowledge and insight in project management, project methodology and in the area of distributed systems. The course runs at 75% of full-time and is rewarded with 15 credit points by running at 50% during the first half of the term and at 100% during the second. The project is to be carried out from planning phase to implementation phase. During the project the students are required to find, use, and, to some extent, invent modern program construction principles and methodologies.

2 Preliminaries

2.1 J2ME

It was decided very early on that the programming language to develop in was Java^{TM}. More specifically Java 2 Platform Standard Edition version 5.0 (J2SE).

On mobile phones however, one is limited to the Java 2 Platform Micro Edition (J2ME), which is similar to J2SE but with highly reduced functionality, although it has some unique features not present in J2SE [J2ME]. J2ME comes in many different forms and flavours: the parts that we chose to use are the Connected Limited Device Configuration (CLDC) 1.1, the Mobile Information Device Profile (MIDP) 2.0 and the Bluetooth API [CLDC,MIDP]. CLDC defines a base set of application programming interfaces (APIs) and a virtual machine for devices with limited resources, like mobile phones, and, coupled with a profile like MIDP, provides a solid runtime environment for Java applications.

Different mobile phone manufacturers decide to implement different parts of the J2ME environment in different mobile phones. This "flaw", or natural technical evolution if you will, determines what code will run where. Many mobile phones therefore have a Bluetooth chip without having the Bluetooth J2ME API.

2.2 Bluetooth

Bluetooth is a technology designed as a means for low-cost, resource-cheap, short range wireless communication [BT]. It operates in 79 different channels in the license-free 2.45 GHz radio band by changing frequency up to 1600 times a second.

Each Bluetooth device has its own Bluetooth chip and with it a unique device identification number. To be able to set up communication the device needs to declare one or more Bluetooth services, much in the same way that a webserver usually has a service at the TCP port 80.

The main issue that was discovered regarding the Bluetooth architecture was the relatively long search times. In the initial game, which only existed briefly as an idea, the intention was to use communication over Bluetooth as the only channel for sending and receiving game data. Unfortunately, this idea fell on the fact that the search times for discovering new Bluetooth devices with a specific service in the surrounding of the device sometimes totalled at half a minute.

2.3 Over-the-air data transfer

The communication between a mobile phone and a computer on the Internet can be done using a socket connection set up in J2ME. The data transmission from the phone must in this case be sent over-the-air (OTA) to a mobile telephony base station which in turn routes the traffic out on the Internet through some system provided by the mobile telephony provider. In a second generation mobile telephony systems (GSM) net, the technology for OTA transmissions is called GPRS [GSM,GPRS]. Third generation mobile telephony systems, like the Universal Mobile Telecommunications Systems (UMTS), provide a seamless interface for data transfer similar to GPRS [UMTS].

GPRS stands for General Packet Radio Service and is a standard for packet data transmission over GSM. The main advantage of GPRS is that a device, such as a mobile phone, may access the Internet from any location as long as it has a sufficient signal strength from the GSM base station. Disadvantages with

GPRS include very high latency and relative low transfer rates. GPRS traffic is often prioritised lower than regular speech on the GSM net. In the common case, ping times average at 300 ms and transfer rates average at 30-80 kbit/s, all depending on phone and operator.

UMTS can be seen as a faster version of GPRS. Transfer rates just below 2 Mbit/s are theoretically possible, but in practice speeds up to 384 kbit/s are expected.

2.4 Phone hardware

The target platform for the game is the Sony Ericsson K750i mobile phone. It is a phone developed for the GSM systems. It features a 1.8 inch TFT screen with 176 x 220 pixels, a possible heap size of 512 kB - 1.5 MB and support for Java technology such as MIDP 2.0, CLDC 1.1 and the Bluetooth API (JSR-82) [K750]. The keypad on the phone consists of the standard 0 to 9 number keys, * and #. It also contains a joystick, three soft buttons, a cancel button and a back button.

3 The Game Three Crowns

3.1 The game world

The game is set in a fantasy world, inspired by glorified tales of the Middle Ages such as the world described by the late JRR Tolkien's The Lord of the Rings. The game world contains three different races: the primitive orcs, the noble elves and the stout dwarves. The story tells of the intense battles between these races. These three races each have their own king, resulting in a total of three crowns, hence the name Three Crowns.

3.2 The avatar

A character representing a player in the game is called an avatar. An avatar has three main attributes being health, magic and strength. Other attributes that are associated with an avatar are credits (in-game money), experience, level, ranking, race and name. The attributes associated with an avatar are often referred to as "stats", short for "statistics". An avatar that engages in battle with another avatar will receive experience and credits at the end of the battle. At a certain amount of experience, the avatar's level will be incremented by one unit. The increase in level will also bring with it a number of distributable points that the player may distribute over the main attributes. The game contains items that can be equipped by an avatar. Determining what items can be used by a given avatar are its three main attributes and the level. To be able to buy items, the amount of credits is also a factor.

3.3 Game play

The game is played by first visiting the game web portal to register an account. The account has to be activated by visiting a link sent to the e-mail specified by the player at the registration. Once activated, the account can be logged into from the webportal and an avatar can be created. To be able to actually play the game, the Three Crowns MIDlet has to be downloaded from the webportal onto a mobile phone. Starting the MIDlet will force it to connect to the game server, prompting the player for the login information specified in the registration process. After the login, the player has a set of options in the main game menu: Play online, Find opponent, My Avatar, Credits and Quit. Quit will exit the application. Credits displays a list of personnel involved in the project. My Avatar will make the MIDlet request and receive from the server the current statistics of the avatar that is logged in on the phone and then present it to the player. Find opponent allows the player to manually enter the name of an opponent to challenge. Play online will make the server run its matching algorithm to find an appropriate opponent for the player's avatar.

The match interface contains picture representations of the two playing avatars, health, magic and strength gauges for both avatars, and the weapon, manoeuvre and spell menus. At the weapon, manoeuvre and spell menus the player can choose which weapon-manoeuvre or weapon-spell combination to use. A time-out is also present at every game round to prevent the game from stagnating. The result of the actions performed by the two avatars is shown at the end of every round. A match ends when one or both of the avatar's health-attribute reaches zero.

3.4 Graphics

By utilizing the J2ME platform we automatically benefit from the fact that the source code will run on several mobile phones, more specifically on mobile phones with screens of different sizes. We realized early on that the graphical user interface of the game could not use fixed points for painting graphics on the screen. Therefore, all non-picture graphics are painted with relative starting points calculated at run-time by the MIDlet by querying the graphics API for screen height and width.

3.5 Web portal

The purpose of the web portals is to complement the MIDlet to make the game social and more complex. At the web portal a player can read information about the game, enter the forum, create a new account or login to an already existing account. Information presented at the portal is sorted and separated into several web pages: "The game", "Highscore", "News", "3C Forum", "About 3C", "Download", "Installation", "F.A.Q", "Team Avatar" and "Contact us". "The Game" presents the background story of the game which explains why the three races are in conflict. "Highscore" presents which three avatars that currently are kings and the ranking of the top 100 avatars. “News” presents the latest news about the game. “3C Forum” leads to the game forum. “About 3C” contains information about “Project Avatar”, the project during which the game was developed. “Download” informs about the download and installation of the game on a mobile phone. “Installation” shows how to install the software phone that makes it possible to play the game on a computer. “F.A.Q” presents the manual for Three Crowns. It explains in detail how to create accounts, play the game and other important information about the game. “Team Avatar” presents Team Avatar, the members of Project Avatar. At “Contact us” contact information for Team Avatar can be found.

When logged in at the web portal the avatar is presented with information such as stats, results (outcome of battles previously fought) and clan. Modifications possible are: to distribute free attribute points, to obtain equipment, manoeuvres or spells, to distribute equipment over the avatar’s different means of storage and to amend the avatar’s autopilot. There is also a forum where a player can log in and discuss in public forum sections or in clan-specific, private forum sections.

3.6 Social aspect

There is a social aspect to the game that we have made some efforts to encourage. A player can challenge any other player by specifying the other player’s avatar name in the Find opponent menu option. From the web portal a player can start a clan. The goal of having a clan is to gather a group of players that can benefit from being a part of the clan. The benefit is achieved through private forum sections accessible only to clan members and a private trading ground where the members can exchange game equipment freely. The forum also have public sections where all players with a game account can discuss everything and nothing.

4 System Description

An overview of the system is shown in Fig. 1. The system consists of a server, a threaded game engine, and game clients.

4.1 Server

The server consists of a database server, a web server and the game engine. The entire server side of the game is currently running on a single computer.

The database server we chose to use was MySQL, the reason to this was that it is free and no other systems seemed to offer any further benefits. The database contains game accounts, including avatar data and match records, as well as all game data such as weapons statistics and abilities statistics. The database also contains all data accessed through the web forum.

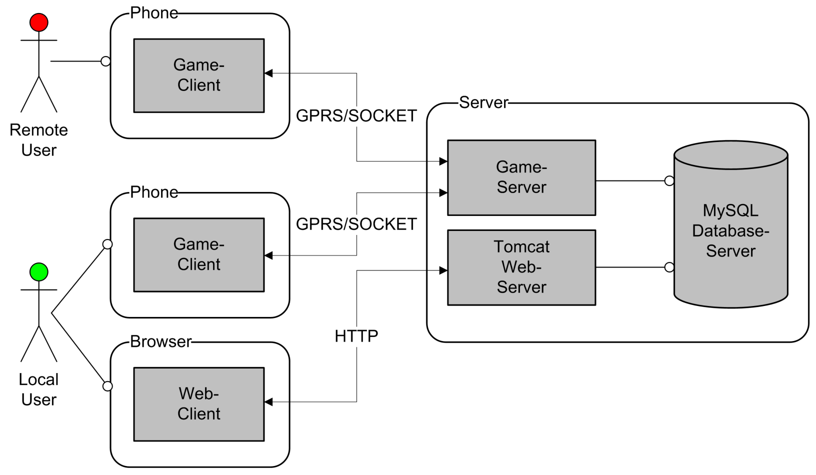

Fig. 1. Overview of system.

The web server is Apache Tomcat. This program allows the use of Java server pages (JSP). The web portal contains information both public and user-specific. The public part holds information that everyone should be able to access and that anyone might want to access such as game ranking, game descriptions, instructions, forums and news. The user-specific part contains account information and everything that has to do with the avatar.

4.2 Game Engine

The game engine is responsible for the game logic and handles connections to the game clients (the mobile phones). The communication between server and client is handled by a custom-made message system using a connection API from MIDP 2.0.

The game engine is the centre of the game. It has a complete view of the game and the states of every client connection and every ongoing match. If there would for any reason be a conflict of opinions, the opinion held by the game engine is the correct one. At the game engine is where the actual game takes place, here is where, for instance, damage calculation is done. The game client is actually little more that an I/O device.

When a user requests to play a match the game engine finds a suitable opponent. An opponents suitability is based on it's level relative to the level of the requesting avatar. The game engine will start looking for an opponent with the same level as the user. If no opponent is found it will expand the search in steps of one level until a opponent is found, but no further than 5 levels away from the users avatar.

After each match the engine will calculate how much experience and credits both players gained and their new rank. When calculating gained experience and credits it will take into consideration which level both avatars had, if one of the avatars won and how much damage the winner recieved. To calculate the ranking a version of the ELO ranking algorithm is used which takes both avatars rank and the match outcome into consideration.

4.3 Game client

The client on the phone is divided into two parts. One handles the communication with the server. The other part is the graphical user interface (GUI), which is responsible for the drawing of the right graphics at the right time. The communication between the two parts is done through the ClientGUI class. ClientGUI consists of a main thread which checks what state the client is in and paints the graphics corresponding to that state. The game state is changed depending on which messages are recieved from the server and the input from the user.

5 Evaluation and Testing

5.1 Requirement verification

In the first few weeks of the project a requirement specification regarding the project and the distributed game system was specified and agreed upon. Involved in the specification were the project supervisors, the product orderer and all project members. We also specified a realization plan intended as a guideline for implementing the system and suggesting optional requirements. In order to verify that the requirement specification was carried out a test plan including a test specification was carried out. The results of the verification tests as well as the test plans were documented, in according to each betarelease's compliance with the requirements.

During the requirement evaluation (and sometimes as early as implementation) we discovered that some of the requirements where too strictly specified or sometimes malformed in such a way that a better product might be implemented if violating a few requirements. In such cases the issues where discussed and the requirements overridden. We also changed the platform from a mainly bluetooth oriented distributed system to an entirely GPRS oriented system after a mutual agreement with the orderer.

5.2 Alpha testing

During the entire implementation phase of the project each module has been tested seperatly directly after implementation. These unit tests where performed in order to test each module with a fresh mind and a direct approach to the might-be-issues. This turned out a big advantage later when testing merged modules interfacing each other. When merging modules we found a big increase in the amount of bugs produced during implementation and decided on using Bugzilla as a bug reporting/handling utility.

The (alpha) test bed consisted of:

- a Debian linux PC with 256 Mb RAM memory, running tomcat, the gameserver and a MySQL database
- a test tool designed to test memory usage when massive simultaneous actions occurs both on the network and database management

A stress test of the server was designed. In order to test memory usage of the java-based gameserver a test tool was written. The tool generated a number of accounts and "simultaneously" logging them in and asking for a game. This test measured the amount of threads that could be allocated while a great number of SQL-querys were issued. Somewhere between 500 and 1000 players proved the memory of our developing servers insufficient, causing SQL memory exceptions.

5.3 Beta testing

Early in the course we decided on recruiting a betatest group consisting mainly of users of the phone intended as target platform, but also with a few anomalities to test the portability of the phone side application. As it turned out many of the phones in the testgroup had not the bluetooth API of J2ME and such tests where more or less left for alpha testing.

The (beta) test bed consisted of:

- a Debian linux PC with 256 Mb RAM memory, running tomcat, the gameserver and a MySQL database
- 53 betatesters whereof 35 active players
- 14 phone models whereof 10 Sony Ericsson phones, mainly K750i and K700i
- the Sony Ericsson WTK2 (wireless toolkit) emulator

The betatesting was carried out in two steps. The first betarelease was issued as soon as there existed a playable version of the game. This was done in order to test the aproximate functionality as soon as possible. A second betarelease was issued when less critical features and functionality was added to the client as well as the gameserver and the portal.

There has been a web-based form so that the betatesters easily could submit bugs experienced while playing. The bug reports have been manually checked and either manually entered into Bugzilla or addressed immediately depending on the severity of the bug.

An evaluation request was issued to the betatesters and the feedback was compiled to an evaluation document intended as guidlines for further development.

5.4 Game balance testing/improvement

We rely on the balance design which has worked sufficiently for testing purposes. Therefor no explicit testing regarding balance tweaking has been issued. It should be noted that in order to introduce the game publicly as a product items and abilites with respective balance to the previous definitions has to be added to make the game as varied as a player might want it.

The game has however been enhanced according to user demands. Such enhancements are user interface for monitoring stats of the player's and the opponent's avatars.

5.5 Project evaluation

The gameclient, gameserver and the portal mostly complies with the specification of the designphase. They have also have features in addition to those of the design. During the entire project we have also collected features that would be nice to introduce into the game outside the scope of the course.

6 Conclusion

6.1 The Project

When starting a large project, organizing and planning are very important. From the start we were encouraged to make as detailed plans and schedules as possible. This seemed easy at first, but after a few months it became clear that our timeplan was far too optimistic in some cases and somewhat pessimistic in other. Because of this we had to revaluate the timeplan throughout the project. The purpose of a timeplan is to make sure the project is completed in time. All team members should know how far the project has proceeded, and how much every member should have produced, at any given time. It is nessecary to have a good overview of what subprojects exist and in which order they need to be completed to create a good timeplan. Finding all subprojects and estimating how much time and effort they require is especially difficult for an inexperienced team as ourselves. One reason this project was as successful as it was, is that we spent a considerable amount of time on the timeplan.

In the beginning of the project, we only had a vauge concept of the final product. This forced us to formulate our own requirement specification (RS), which has both advantages and disadvantages. The obvious advantage is that we could include things we found interesting and wanted to learn more about, e.g. the bluetooth part of the game. Another good thing is that we could design the RS such that we felt secure in our ability to realize all requirements. On the other hand writing your own RS can lower the standard of the project, meaning that some difficult requirements may have been left out. Additionally, the client does not have as much insight in the project and therefore he does not have the same ability to affect the project outcome. Finally, creating a RS takes a lot of manhours from a projects allocated time.

There are many factors that determine if a project will be successful. One of them is having good relations within the developing team. When this project started we were eight individuals whith little or no experience working together with the others. One purpose of this is to allow students to take part of a life like working environment. Our approach to unite the team was first make sure that everyone was at work at the same time as much as possible. Group meetings was also an important factor, allowing everyone to speak his mind freely.

6.2 The Game

The entire game is made in Java, partly for simplicity and partly to be able to use the same code in both the midlet and the server. When the first game prototype was finished we realized that the server running java was not able to handle a large amount of concurrent connections. Had we considered this problem earlier, the server would most likely have been written in some other language, possibly Erlang.

References

[J2ME] "Java 2 Platform, Micro Edition (J2ME) Overview", http://java.sun.com/j2me/overview.html, (2006-01-31)

[CLDC] "Connected Limited Device Configuration (CLDC)", http://java.sun.com/products/cldc/index.jsp, (2006-01-31)

[MIDP] "Mobile Information Device Profile (MIDP)", http://java.sun.com/products/midp/index.jsp, (2006-01-31)

[BT] "Bluetooth - Wikipedia, the free encyclopedia", http://en.wikipedia.org/wiki/Bluetooth, (2006-01-31)

[GSM] "Global System for Mobile Communications - Wikipedia, the free encyclopedia", http://en.wikipedia.org/wiki/Global_System_for_Mobile_Communications, (2006-01-31)

[GPRS] "General Packet Radio Service - Wikipedia, the free encyclopedia", http://en.wikipedia.org/wiki/General_Packet_Radio_Service, (2006-01-31)

[UMTS] "Universal Mobile Telecommunications System - Wikipedia, the free encyclopedia", http://en.wikipedia.org/wiki/Universal_Mobile_Telecommunications_System, (2006-01-31)

[K750] "K750", http://developer.sonyericsson.com/site/global/products/phones/k750/p_k750.jsp, (2006-01-31)