Geo Wars – The development of a location-based game

Geo Wars – O desenvolvimento de um jogo baseado na localização

João Tiago Pinheiro Neto Jacob

Faculdade de Engenharia, Universidade do Porto
joao.jacob@fe.up.pt

António Fernando Coelho

Faculdade de Engenharia, Universidade do Porto / INESC Porto
acoelho@fe.up.pt

Resumo

Com o aparecimento de novos dispositivos móveis como smartphones, tables e PDAs, surgem também novos paradigmas de interação para utilizadores e programadores explorarem. Estes dispositivos permitem que sejam criadas aplicações únicas graças ao vasto número de sensores e mecanismos de entradas que possuem (como GPS, acelerómetros, entre outros). Assim, experiencias ubíquas e personalizadas tornam-se possíveis. Jogos móveis baseados na localização não se limitam a copiar mecanicas de jogo já existentes em jogos de consola/computador. Pelo contrário inovam, permitindo experiencias únicas e personalizadas. Este artigo apresenta um jogo móvel baseado na localização: desde a fase de desenho e implementação à fase de testes do mesmo.

Palavras-chave: Jogos Móveis; Serviços de localização; GPS; Dispositivos móveis.

Abstract

As new, more powerful, mobile devices, such as smartphones, tablets and PDAs arrive, so do new ways of interaction for the users and developers to explore. These devices allow for the creation of never-before-seen applications, using the broad range of inputs and sensors these devices posses (such as accelerometers, light sensors, compasses, GPS sensor among others), that allow for more ubiquous and personalized experiences. Mobile location-based games are such applications, not being limited to copying previous console/computer games and their concepts, but rather inovating them so as to provide different and unique gaming experiences.

Keywords: *Mobile games; Location awareness; Location-based services; GPS.*

1. Introduction

All software that is created follows a given methodology. The interactive entertainment industry, more commonly known as game industry is no different. Some methodologies, such as sashimi or the spiral/iterative model (Araújo & Roque, 2009) help determine what aspects of the project should be altered, added or removed by analysing the end result (or, in case of the sashimi model, the result of a given phase). As important as this is in the software industry, its importance is even greater in video games, as these are often complex and unpredictable. Even more unpredictable are the location-based games, as these games rely upon location data, such as the player's real position, as a means of input or as a means of generating or using game specific content. As such, the player's game experience may differ, not only from player to player, but from location to location, expanding the game's possibilities and longevity. Since these games rely upon location services to work (or, in their simplest of forms, just a positioning system such as a GPS module) they are almost exclusively available in mobile platforms, since these are the most prone to be more up to the task, as they have both the necessary services, and as they are by definition mobile, the game can be played easily in several locations or even while the player is moving himself. However, certain problems are due to arise from the usage of such dynamic information as location based one. Location based games become often unplayable in situations where the location services are somehow inoperative (such as inside buildings when the GPS module is incapable of getting a satellite fix) or when the location-based data is unavailable or poor for that particular location (as when there is no information the game may use for a specific location, or the information provides a bad user experience). Also to be considered is the gamer's physical condition.

This paper presents a location-based game called Geo Wars and all of its phases of design, development, implementation and testing. The following chapters will cover the actual state-of-the-art of location-based games, some of the issues present in these games, the game's mechanics, the game's architecture, the game's development and finally, the conclusion and the future work of this project.

2. Location-Based Games

A Location-based Game is a game that uses the player's physical location (or any other location) as a means of input or to generate or access location-based information. These games are almost exclusively available on mobile platforms [MGD]. Location-based games haven't been around for too long. In fact, they would only see any commercial use in 2002 with the arrival of Botfighters [BotFig], the first pay-per-locate GPS game. The concept was simple: each player was a robot with the mission of destroying the enemy robots. For each kill the player would be awarded credits which could be spent on the game's website to further improve his killing machine. In order to play the game, the player would move around the city, scanning for enemy robots (which would be other players with the same goal). When an enemy robot was in a 500 meter radius of the player's position, an SMS based exchange of fire would commence. The effectiveness of the shoots would be greater as the distance between target and shooter shortened. This allowed for a kind of tag game, as players often would engage in urban pursuits, chasing and running away from attackers. With the birth of of this genre only roughly eight years ago, location-based games have now gained a considerable popularity. For an instance, Geocaching [GeoCa], probably the most popular of geocaching games, claims to have more than a million registered users. The idea behind it is even simpler: a player stashes a "cache" (with contents in accordance with the geocaching's rules) anywhere in the world and shares it's approximate location with other geocachers around the globe. Other players will attempt to discover this cache by solving some riddles and exploring the general area the cache is known to be. After finding the cache they may exchange an item, part of the cache's content, with any owned item. This game is widely played not only due to its simplicity and likeness to treasure hunting games, but also because it doesn't necessarily require a GPS capable device, as players may be already familiar with the general area of the cache prior to going to find it. After finding a cache the player may return to the game's website in order to update his track of found caches and to provide feedback to the player that created that particular adventure by stashing the cache.

Of course other games, although using the physical location of the player, need to be played with a mobile location-capable device. Pac-Manhattan [PacMan] is such a game. In it, players will have to enact a classic Pac-Man game. In order to do so, ten players are required: one to be Pac-Man, four other to be ghosts, and the remaining five to control and coordinate the actions of the others via mobile phone. As the Pac-Man player wanders around the streets of a real city, the player responsible for controlling him will guide him through the streets, avoiding the ghosts and, of course, eating pellets. The other coordinators will attempt to guide the ghost players in order to trap (or if Pac-Man has eaten a big pellet, to escape) the Pac-Man player. Only the coordinators have, in their mobile application, the full picture, the maze and all the five player's positions in the game. The real five players have no picture of the game, other than being themselves on the terrain.

As location-based games gained popularity, several platforms of mobile games emerged, with two having a particular success. One of them is Groundspeak [GroSp], the platform that contains not only the Geocaching game, among others, but also the Whereigo [WiGo] tool, a tool for creating and playing GPS-enabled games [WiGo]. The other, Locomatrix [LoCo], currently in quick expansion, offers two games: Treasure hunt, a geocaching like location based game, with a more virtual component, as the player has to use pictures to solve riddles and find places, and fruit farmer, a game where the player must collect "fruits" around him by moving around in the real world, avoiding also having these fruits stolen by other farmers. As it can be seen, location-based games have come to be quite a success from their origins in the early 2002. These games' communities grow bigger every day, with increasingly more new location-based games being added to these platforms.

3. Issues in the Development of Location-Based Games

The core of the mechanics of a location-based game is the location component and some other derived issues. The problem is rarely how to integrate location-based information into a game's logic, but rather the accessibility of that information and the unpredictability that follows its integration. Most location-based games rely upon the GPS module available on smart phones in order to provide the player's physical location. This means that the availability of that information depends on the GPS signal, a signal that may not always be present when needed. This implies that the location-based game may not even have the information of the player's location that it needs and thus the game is rendered useless or unplayable in most cases. Even when the signal is available, sometimes there is no location related content available for the location the player is at to be used in the game, crippling or making the game impossible to play. Once again, even if considering that the signal is present and the content is available, due to the unpredictability of the quality of that content there is usually no guarantee that the game can provide a good gaming experience. As such, location-based games are games that have a huge unpredictability concerning the gaming experience, capable of surprising the player in both good and bad ways. Another issue, for those games that use the player's own movement as an input, is the lack of consideration of the player's fitness and athletic skills. As such, a game that requires the player to move a distance of a mile in the real world may be an excruciating task for the unfit gamer or may be excessively easy and boring for the more athletic one. And so, another problem arises. Since the game uses not only the game specific skills usually required in most games but also requires physical movement, the gaming experience and level of challenge the game poses to the player is also connected with the different ad equability of this component for each and every player. The solution for this issue would be a dynamic balancing mechanism that would take the player's current physical activity into account and would regulate the game's pace accordingly. Additionally, the inclusion of tutorial levels, levels that not only teach the player the dynamics of the game but also calibrate the game's difficulty using the player's performance, is also a valid solution.

4. Game's Mechanics

Geo Wars is a location-based game, which premise is based upon tower defence strategy games. The idea behind the development of this game was to determine what were the limitations and issues in a location-based game, and overcome those issues using an adequate and justified methodology. In these types of games (tower defence strategy games), the player must defend a base or position from invading enemies. These enemies move along paths that the player cannot obstruct with buildings. The player may build towers, each with different capabilities such as weapon's range, firepower and other variables that are meant to hinder or halt the enemy invasion. Geo Wars takes this premise further by using real life maps as a base for this game. As such, the enemies can only move around streets present on the real map. The map chosen for each game is the map representing the general area, usually in a 3 or 4 block radius, the player is located when playing the game. In order to make it even more interesting, the target of the enemy forces is the player position itself.

As such, the player may choose to flee, moving in the real world in order to do so, as its avatar is moved only via GPS input, or build defensive towers in the hope of fending off the waves of enemies until the game time is over. Moving around the map is rewarded by random treasure chests that pop up randomly around the map's streets which the player may pick up by moving over them. The enemy units will attempt to fire at the player, but each of these units is gifted with a unique and distinct A.I., as well as other limitations. This A.I., developed specifically for this game, is composed by 3 distinct phases for all units: target acquisition, moving and firing sequence. These 3 phases however, vary among different types of units. For an instance, while a Tank will acquire a target following a priority list, preferring first Machine Gun Towers, then Rocket Towers and so on (meaning that if there are any Machine Gun Towers in the map it will attack them, and always disregard subsequent units in its target list), a Boat will always target the player.

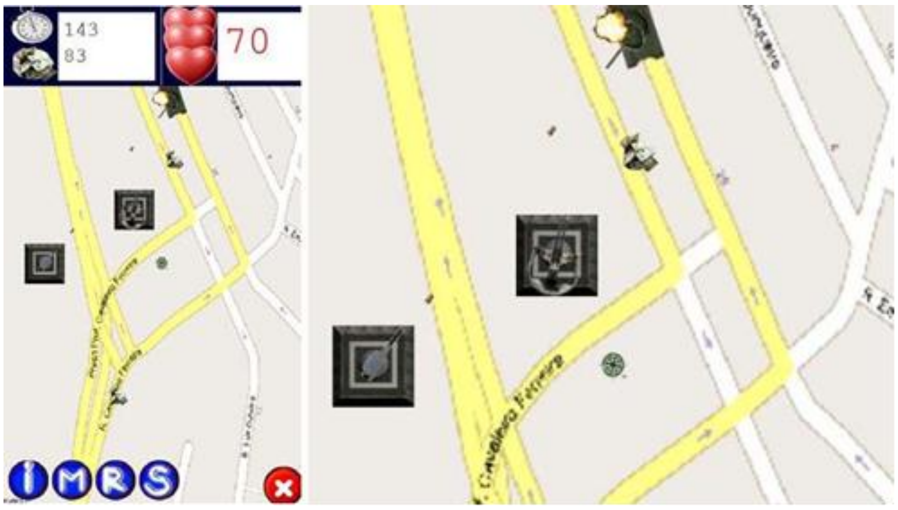

Figure 1 - ***Geo Wars look and feel***

Figure 1 depicts a tank following a road in order to get closer to the player's location while firing back at a tower. The player's location, marked by a green compass in the centre of the screen shows the real location of the player's position in the game. Also scattered around the map's streets are bags of money that the player may or may not pick by moving over them in order to collect extra money. The overhead GUI depicts the player's current health, money and match time left. On the bottom of the screen lies a succession of buttons that allow the player to deploy different types of towers. Finally at the bottom right corner lies the pause/main menu button, a feature much needed in a game that requires the tiresome task of making the player move around the real world.

Also implemented was the usage of real time local weather that influences the game, as good weather boosts the player's towers as well as its enemies, while bad, cloudy or rainy weather, penalises the invading forces' speed and weapon range. Additionally, as an extra, it is also given to the player the opportunity of tweaking the game's settings, such as manually inserting the address of the place of where they would like to play the game. This way, the player can enjoy a custom tailored game experience, useful when the player is playing the game in a place where the gaming experience might be rather poor, for an instance, when camping or travelling. Also, the player may wish to disable the usage of the GPS module altogether, allowing the player to enjoy a location-based game (although obviously not based on his position) even in places where he would be unable to play, such as inside buildings. The player may even wish to broaden the area of the game's map in order to better fit his gamming skills and his physical abilities, by selecting the zoom level of a map. This affects how fast the player's avatar moves, as a higher zoom level means that the game's screen depicts a smaller area, and thus the player's speed in pixels is increased. So, while the player moves faster, the other units move at a similar, constant pace, meaning that the player may customize the game to meet his physical abilities. Additionally, the player may also chose to play a longer or shorter match by introducing the number of seconds he wishes the match to last at most. Each and every game in Geo Wars only ends when either time runs out or the player dies. The final score for each game takes into consideration how much money was he left with, how long did he survive, and how much life was the player left with.

5. Methodology

For the development of the Geo Wars game and all the above mentioned components of the game, a particular approach was used. This approach, aimed to be as general as possible, so as to be applicable to any other location-based games that might be developed. Before the development of the game itself or any of its core components, a brief investigation of the location-based game's history was conducted. The goal of this research was to determine the type of environment these games are played in, their limitations, their players' stereotype, and the risks associated with their usage and which content is usually used as location based content. This investigation was the basis for the creation of a general architecture of a location-based game, a solution that aims to correct or avoid most problems related to location based games, while being general enough to be implemented in virtually any system, architecture or game concept. The game itself is based upon the principles and methodologies here listed, although in some cases a more specific solution was chosen due to the nature of the game, the underlying system, or the device itself. However, the core principles were used. The design and implementation of the concept followed a Waterfall methodology variant called the Sashimi model, as the project's phases overlapped, allowing to adapt the design phase of the project as some features in some components gave their feedback, meaning that during the development of several components of this project, the flow of it could go back and forth [MeDeOr]. Furthermore during the solution's testing, it was clear that some useful features were being put aside, and so using this insight the design and architecture of the game was adapted so as to include these features. Thanks to the flexibility provided by the Sashimi model, no phase of the software's cycle of development was fully completed before advancing to the next phase, allowing to move back and forth along the cycle, using the feedback each phase provided in order to better complete the blanks in the other phases. This methodology proved to be quite useful in a situation where a phase couldn't be fully completed due to lack of complete knowledge of the problem. As the development of the game's concept grew, so did the requirements of what the location module of the game should do, adding to it a weather web service access, and a more typed parsing of a map's structure as the game's features required.

6. Game's Architecture

Besides the mobile application, a website, database and web service were also created. Using the website, any user may build a customized game from scratch and share it with other users. This way, when a player logs in he can access a list of custom-made games. When finishing any of these games, the user's score is also submitted online in order to share it with other players. All this information regarding scores, user details and predefined saved games is stored in the database. The mobile application accesses this database via the web service, in order to assure safety. The website, however accesses this database directly.

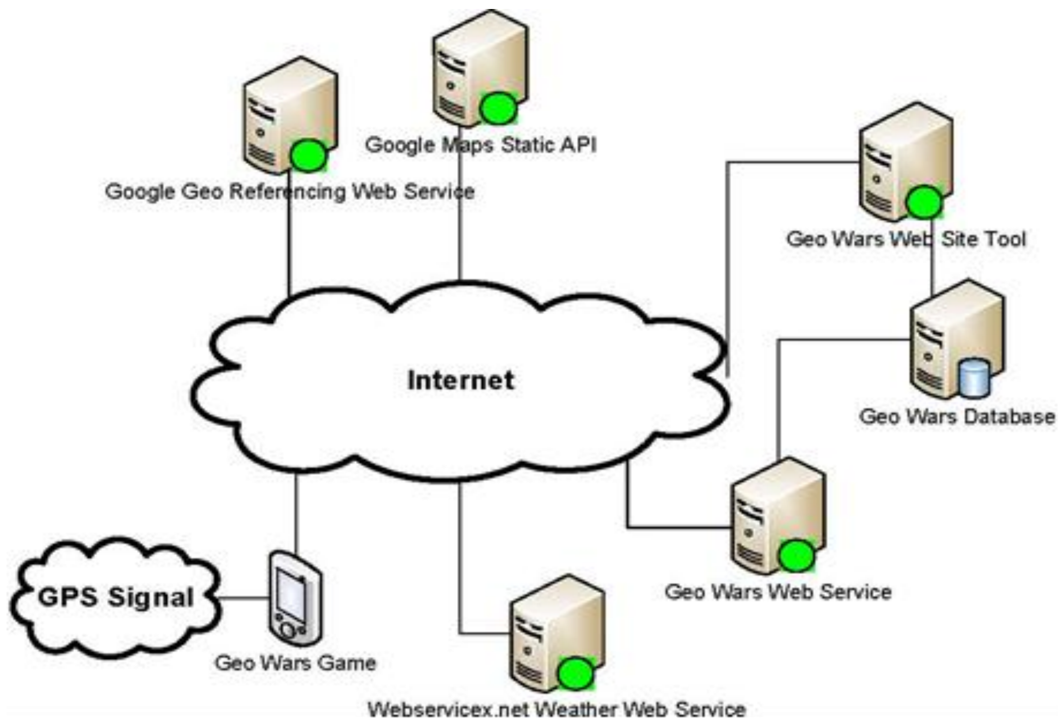

Figure 2 - ***Geo Wars physical architecture***

As it can be seen in the figure 2, after acquiring the GPS signal, the Geo Wars application accesses the Internet to obtain the Map image of the player's or any other given location using the Google Maps Static API. If the user desires to use the weather for his current location the game will attempt to use the Webservicex.net's Weather web service, after converting the player's location coordinates into its respective city and country components by using Google's Geo Referencing web service. In order to access stored games and in order to submit scores, the game will access the Geo Wars web service, an interface to be used for safe and easy to use access to the Geo Wars Database. If the mobile device also possesses a browser, the player can use this browser to access the Geo Wars web site tool so as to create and browse through his games and scores.

7. Game's Development

The game was developed for Windows Mobile based devices, using both the .Net Compact Framework 3.5 and the Direct3D Mobile API. Before the creation of the game's logic and its elements themself there was a need to develop a small yet solid location-based game framework, generic enough to be applicable to any location-based game concept. As such, it was necessary to create Sprite primitives (as the game is viewed in a orthographic perspective) and their respective picking and collision routines. Also, due to the mobile devices lack of large amounts of available RAM, it was needed to do some instantiation of textures in order to prevent Out Of Memory Exceptions, as each process in windows mobile is only reserved with around 20 MBs of RAM. Furthermore, since these games usually rely upon a lot of trigonometry, this solution also computes sines and cosines for 360 values of angles while the user is at the main menu of the game in an attempt to cache them and to be easily accessible when needed, thus reducing processing power needed while in the game. The game architecture also integrates the Windows Mobile Intermediate Driver, a solution that allows the developer to easily and readily access the GPS module in order to read its values. The game, after receiving valid longitude-latitude NMEA data (a specification that the National Marine Electronics Association developed for data exchange) or the user's Google Maps style address, attempts to access location data based upon it. However, not only is the Location module of the game used prior to the game's start, it is also used every second or so while the game is occurring in order to correctly move the player around the map (of course, this feature only works if it has been enabled by the player beforehand).

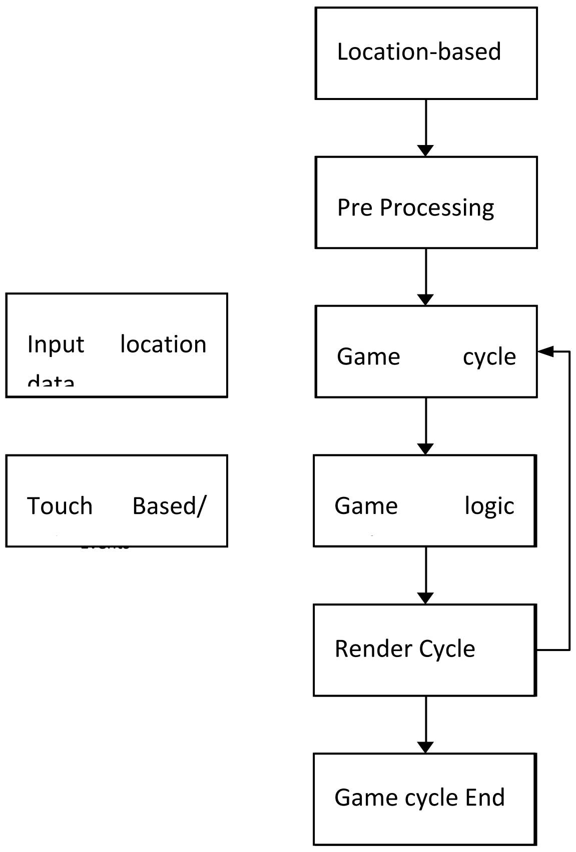

Figure 3 - ***Generic Location Based Game Flow Chart***

In the schematic in figure 3 it can be seen that, while some of the needed location-based data is loaded before the start of the game's cycle, during the pre-processing phase, another thread, parallel with the game itself, will continually determine if the player's real world position has changed, and if it has, that same thread will change some of the location-based information. At each game logic cycle this location-based information is integrated with the game's logic. Also, touch-based events are captured in a different thread, since the user builds towers and interacts with the interface by touching the screen. The Location module of the game uses the Google Maps Static API to get the image that represents the location of the given address or the GPS values. Afterward, that very same image is analyzed in order to determine what are the parts of that map that are to be considered roads, green spaces, water zones, buildings and other elements. This all happens just during the pre-processing phase of a game. It is only done once per game. This way, it is possible to integrate that information with the game's logic. It is important to note that while the Location module of the game only accesses and stores location data related information, such as player's position, map nodes and their respective types, among other things, it is the Game module that accesses this information and integrates it into the game logic accordingly. The images in figure 4 show the map that is obtained from the input “Braganca, Portugal” and what information is retrieved from that map. White areas represent areas that can be used by an enemy soldier or tank to move around. Red areas are the only areas where a tower may be placed by the player.

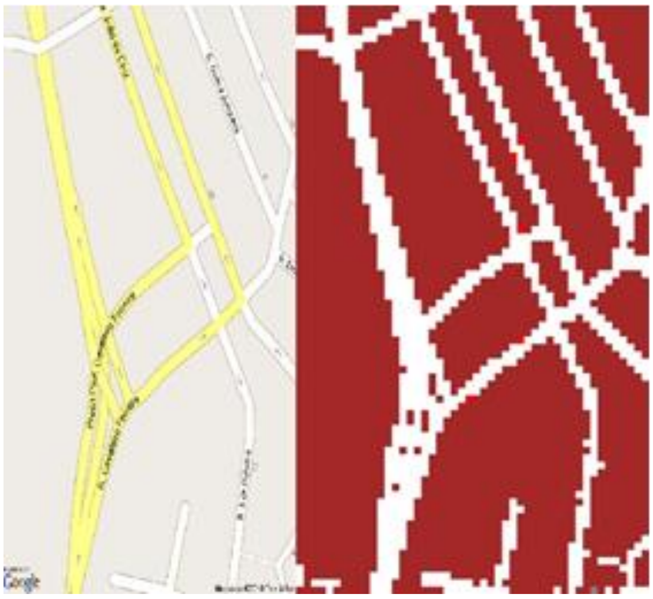

Figure 4 - ***Original game map and visual representation of the information retrieved from it***

It is noticeable that the rightmost image in figure 4 depicts the information retrieved from the other image in lower resolution. In fact, information retrieved from the map is not obtained per pixel, but rather by first shrinking the original image and then parsing. This allows the representation of the map’s structure to be simplified, so that the path finding algorithm (in this case a typical A* algorithm, with 8-connectivity and usual distance based heuristics) to use this simpler map with good results and with increased speed for an overall acceptable precision.

8. Conclusions and Future Work

During the course of this project, it became notorious that some limitations in location-based games were difficult if not impossible to overcome completely due to their natural unpredictability. As such, some of the mentioned issues were not solved as much as they were avoided. An example of this is the lack of GPS coverage in buildings that was overcome with the feature of allowing the player to play a game that is not based on his location. This doesn’t constitute a solution for the problem but an alternative to not being able to play a location-based game at all. It also became evident that balancing a location-based game is rather difficult as some players may find the game too easy or too difficult from location to location. Even when taking that into account it is difficult to measure how hard or how easy the game should be based on the location related information. As such, balancing issues of the game were, by the end of the testing phase of development, considered to be the most difficult to solve. However, and thanks to the methodology based on the agile Sashimi model, that allowed to adjust requirements, features and the design of the solution even as other phases of the project went on being developed, many solutions for issues of location-based games that were only spotted during the implementation phase of the project were included in the game’s design. Additionally thanks to the creation of a general location-based game solution (the previously mentioned framework, the base upon Geo Wars was implemented), the game’s final concept was only elaborated and completed long after a simpler and different location based game was implemented using the very same general design, proving that the generic approach to the location-based games was possible through the said framework.

The game Geo Wars has had a good acceptance in the mobile gaming community as it can be seen in this *xda-developers* thread used for testing and gathering of feedback to be used in subsequent iterations (http://forum.xda-developers.com/showthread.php?t=694276). So much that the framework is being ported to 3D so that Geo Wars may also be a 3D game. Prototype versions for both 2D and 3D are also available for download in said thread as well as a small video depicting a Geo Wars match. Many of the testers of the game asked for a multiplayer version of the game. Alas, the framework, as of now, is incapable of allowing such a feature, although it will most likely be altered in order to accommodate that possibility.

9. Acknowledgments

This work is partially supported by the Portuguese government, through the National Foundation for Science and Technology - FCT (Fundação para a Ciência e a Tecnologia) and the European Union (COMPETE, QREN and FEDER) through the project PTDC/EIA-

EIA/108982/2008 entitled “3DWikiU – 3D Wiki for Urban Environments”.

10. Bibliographical References

[MDA] Hunicke ,Robin, LeBlanc ,Marc, Zubek ,Robert (2004) “MDA: A Formal Approach to Game Design and Game Research “, Challenges in Game AI, 2004

[GDE] Novak, Jeannie (2008), “Game Development Essentials”, Delmar Cengage Learning

[FLBS] Steiniger, Stefan, Neun, Moritz Edwardes, Alistair (2006) “Foundations of Location Based Services” in “Lecture Notes on LBS”, 2006.

[LoCo] Locomatrix, 2010. http://www.locomatrix.com/about.html , 10/2009.

[GroSp] Groundspeak 2010 http://www.groundspeak.com/ , 10/2009.

[GeoCa] GroundSpeak.Geocaching, 2010.http://www.geocaching.com/ , 10/2009.

[GPSGa] GPSgames.org.GPSGames.org, 2009 . http://www.gpsgames.org/ , 10/2009.

[MinWar] GPS Games. Minute War, 2006. http://ultimategps.com/gps_fun.html , 10/2009.

[Shutter] GPS Games. Shutterspot http://www.gpsgames.org/index.php?option=com_wrapper&wrap=Shutterspot , 10/2009.

[PhoTag] Navigadget. “Take down your targets! A gps phone tag game.” http://www.navigadget.com/index.php/2006/04/04/take-down-your-targets-a-gps-phone-tag-game , 10/2009

[WiGo] GroundSpeak, WhereIGo, 2008. http://www.wherigo.com/, 10/2009

[PacMan] Pac Manhattan http://www.pacmanhattan.com/about.php, 10/2009

[BotFig] Dodson, Sean (2002) “Ready, aim, text”, The Guardian, http://www.guardian.co.uk/technology/2002/aug/15/electronicgoods.games, 5/2010

[MeDeOr] Araújo, Manuel, Roque, Licínio (2009) “Uma proposta metodológica para organizar o desenvolvimento de jogos originais”, VideoJogos 2009

[MGD] Joselli, Mark, Clua, Esteban (2009) “Mobile Game Development: A Survey on the Technology and Platforms for Mobile Game Development”, VideoJogos 2009

[Sashi] Rising, Jim (2009) “Sashimi Waterfall Software Development Proccess”, Managed Mayhem, http://www.managedmayhem.com/2009/05/06/sashimi-waterfall-software-development-process/ . (last accessed on June 14th, 2009)